APPLICATIONS OF BAYESIAN FILTERING IN WIRELESS NETWORKS: CLOCK SYNCHRONIZATION, LOCALIZATION, AND RF TOMOGRAPHY

A Dissertation
Presented to
The Academic Faculty

by

Benjamin Russell Hamilton

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2012

APPLICATIONS OF BAYESIAN FILTERING IN WIRELESS NETWORKS: CLOCK SYNCHRONIZATION, LOCALIZATION, AND RF TOMOGRAPHY

Approved by:

Dr. Xiaoli Ma, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Robert J. Baxley, Co-advisor
Information and Communications Laboratory
Georgia Tech Research Institute

Dr. Justin Romberg
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. G. Tong Zhou
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. John Barry
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Jun Xu
College of Computing
Georgia Institute of Technology

Date Approved: April 20, 2012

ACKNOWLEDGEMENTS

There are uncountably many people who have contributed in some way to the body of research represented by this dissertation or who affected me personally to educate, inspire, and challenge me to persevere to the point I am at professionally. I describe only a small portion of them below.

I would like to thank my advisors, Dr. Xiaoli Ma and Dr. Robert J. Baxley. They brought me into the world of academia, taught me to think as a researcher, supported and guided me through many of my projects, and gave me the confidence to perform independent research.

I would also like to thank the other members of my Ph.D. dissertation reading committee: Dr. Justin Romberg, Dr. G. Tong Zhou, and Dr. John Barry. Their support and suggestions have greatly improved the quality of this dissertation.

This work would also not be possible without those who collaborated with me on my research: Dr. Stephen Matechik, Dr. Brett Walkenhorst, John Kleider, Hayang Kim, Dr. Qi Zhao, Dr. Jun Xu, Dr. Min-Te Sun, and everyone else who has supported my research.

I would also like to thank my wife, Dr. Lei Hamilton, for her support and encouragement as well as her patience with me through the more difficult portions of my research, my parents, David and Donna Hamilton, for always pushing me to achieve, and the many educators, without whom I would not be who I am today: from the professors at Georgia Tech and Auburn University to the teachers at Auburn High School and the other schools in the Auburn public school system.

Thanks also go to the Georgia Tech Research Institute, whose "GTRI Shackleford Fellowship" provided me not only a secure financial foundation to focus on my research, but also access to the amazing professionals and laboratory facilities that made this work possible.

Additionally, the research in Chapter 3 was sponsored (in part) by the Air Force 46th

Test Systems Support Squadron, Eglin AFB, FL under the Command, Control, Communications, Computer, Intelligence (C4I), and Munitions Test Improvement Contract III (CIMTIC III).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In this work, we investigate the application of Bayesian filtering techniques such as Kalman Filtering and Particle filtering to the problems of network time synchronization, self-localization and radio-frequency (RF) tomography in wireless networks. Networks of large numbers of small, cheap, mobile wireless devices have shown enormous potential in applications ranging from intrusion detection to environmental monitoring. These applications require the devices to have accurate time and position estimates, however traditional techniques may not be available. Additionally RF tomography offers a new paradigm to sense the network environment and could greatly enhance existing network capabilities.

While there are some existing works addressing these problems, they all suffer from limitations. Current time synchronization methods are not energy efficient on small wireless devices with low quality oscillators. Existing localization methods do not consider additional sources of information available to nodes in the network such as measurements from accelerometers or models of the shadowing environment in the network. RF tomography has only been examined briefly in such networks, and current algorithms can not handle node mobility and rely on shadowing models that have not been experimentally verified.

We address the time synchronization problem by analyzing the characteristics of the clocks in small wireless devices, developing a model for it, and then applying a Kalman filter to track both clock offset and skew. In our investigation into RF tomography, we present a method using a Kalman filter which jointly estimates and tracks static and dynamic objects in the environment. We also use channel measurements collected from a field test of our RF tomography testbed to compare RF shadowing models. For the localization problem, we present two algorithms incorporating additional information for improved localization: one based on a distributed extended Kalman filter that combines local acceleration measurements with signal strength measurements for improved localization, and another that uses a distributed particle filter to incorporate a model of the channel environment.

CHAPTER I

INTRODUCTION

Bayesian filtering is a powerful technique for estimation of a random process from observations [3]. When the probability distributions of the observations, the process, and the conditional distributions are known, Bayesian filtering obtains the optimal estimate of the state distribution of the random process. The general Bayes filter is impractical in most applications due to its complexity; however, if some assumptions are made about the distributions of the observations and states, several practical simplifications exist. Such simplifications include [4] techniques such as Kalman filtering [5] and particle filtering [6]. These techniques have demonstrated their utility in applications in a wide range of fields.

Recent advances have enabled the creation of small, mobile, wireless devices. Such devices have been proposed for applications ranging from enhancing personal communications to environmental modeling. Networks of large numbers of small, cheap, mobile wireless devices have shown enormous potential not only in civilian applications ranging from intrusion detection to environmental monitoring, but also have military applications such as monitoring friendly forces, providing surveillance of both opposing forces and terrain, and detection of chemical, biological or radiological hazards [7].

These networks pose particular challenges not found in traditional networks. The number of wireless devices, lack of fixed infrastructure, and dynamic nature of these networks makes traditional configuration and management techniques seem sluggish. Instead, research has focused on algorithms for the network devices themselves to dynamically form a network. Although several works [8–14] have dealt with the problems of routing and data distribution in the network, many applications require additional services such as time synchronization and device localization. Traditional techniques such as GPS may not be available due to cost or environmental factors. Additionally the capability to obtain images of the network environment and track moving objects using existing receive signal strength

(RSS) measurements through RF tomography could greatly enhance the sensing capabilities of such networks.

While these challenges have been addressed by several works, they all suffer from limitations. Existing time synchronization methods were not designed for small wireless devices with low quality clocks, and so are not energy efficient. While the RF tomography problem has been examined in several works, current algorithms use RF propagation models that have not been experimentally verified, and do not consider effects of node mobility in wireless networks. Existing localization techniques do not consider the additional sources of information available to nodes in the network. Small wireless devices often contain accelerometers whose measurements could be used to improve location tracking. Further, improvements in channel estimation such as models of the shadowing environment in the network could also be leveraged to improve localization accuracy.

In Chapter 2, we address the time synchronization problem by analyzing the characteristics of the clocks in small wireless devices. We use this analysis to develop a novel clock model that more accurately reflects the properties of the low quality clocks used in small wireless devices. We then apply a Kalman filter to track both clock offset and skew, the rate the offset changes. We evaluate our algorithm both theoretically and through simulation to demonstrate its performance.

We examine the RF tomography problem in detail in Chapter 3. We describe the theoretical basis behind RF tomography and how it relates to traditional tomography techniques. We then propose a new RF tomography algorithm, RETINA (RF Exploitation for Tomographic Imaging and Non-cooperative Analysis), which uses a Kalman filter to jointly image both stationary and mobile objects even when the wireless nodes are mobile. We compare RETINA to existing algorithms through simulation and demonstrate its improved accuracy. Further, we describe our RF tomography testbed and use the measurements from a field test to compare three different RF propagation models and show that our novel model, the Inverse Area Ellipse model, is the most consistent with the observed measurements.

In Chapter 4, We investigate the RSS-based localization problem with a focus on incorporating additional information such as measurements from accelerometers and models

of the wireless channel. We present two distributed localization algorithms. The Hybrid Location Tracking algorithm uses a distributed extended Kalman filter (EKF) to incorporate acceleration information. The Shadowing Assisted Localization (SAL) algorithm uses a model of the wireless channel in the network, such as the models we discuss in Chapter 3, along with a distributed particle filter. We demonstrate the performance of each of these methods through simulation and show that the use of these additional sources of information significantly improves localization accuracy. Additionally, we investigate a strategy to modify these algorithms to improve accuracy and convergence time through the additional exchange of location estimate variance information. Results from simulation confirm that incorporation of this strategy is advantageous in both of the proposed localization algorithms.

CHAPTER II

NETWORK TIME SYNCHRONIZATION

The availability of an accurately synchronized clock enables and enhances a wide range of applications in distributed environments. For example, Internet measurements relying on either passively monitoring network events (e.g., packet loss) or actively probing network conditions (e.g., end-to-end delay and loss rate) implicitly require a common notion of time among all participating measurement points. Another example lies in Wireless Sensor Networks (WSNs). Sensor network applications need a common notion of time for precise data integration and sensor reading fusion. Clock synchronization is also essential in network and communications protocols such as TDMA medium access scheduling, node sleep scheduling, and scheduling for directional antenna reception.

2.1 Problem Statement

Network time synchronization is simply the problem of setting two or more clocks with the same notion of time, and performing updates to ensure this continues to occur. This problem becomes complicated however, when the characteristics of the network and clocks themselves are considered. Oscillators in clocks suffer from skew, drift and jitter. all of these cause the clocks to progress somewhat erratically. Information sent over a network is subject to random, variable delays which add significant measurement noise to time measurements.

Clock drift refers to the phenomena where a clock does not run at the correct speed compared to the actual time. The phase noise in oscillators is an important component of clock drift. Because phase noise is random, the clock drift is also random. Clocks often drift differently depending on their oscillator quality, the exact power they get from the battery, temperature, pressure, humidity, age and so on. Thus the same clock could have different clock drift rates on different occasions. Usually the instantaneous clock drift rate is called *clock skew* and the time difference with the actual time is called *clock offset*.

Clock synchronization relies on the transmission of time measurements over the network.

The delay between the creation of the time measurements and their arrival at their destination is caused by a variety of independent factors, and can often be thought of as random. Two measurement models are generally employed for time synchronization: sender-receiver synchronization and receiver-receiver synchronization. In sender-receiver synchronization, the receiver tries to synchronize their clock with the node that transmitted the measurement. In receiver-receiver synchronization, two receivers synchronize with each other by using another sender's transmission as a synchronization point. In either case, the network introduces delay into the time measurements. Constant symmetric components of the delay can be estimated from the packet round trip time and removed. Likewise predictable time delays can be eliminated (such as propagation delay when node positions are known). In general, the time-critical path in a sender-to-receiver synchronization consists of four factors [15]: i) the time for message construction and sender's system overhead, ii) the time to access the transmit channel, iii) propagation delay, and iv) the time spent by the receiver to process the message. In contrast, a receiver-to-receiver synchronization is only impacted by iii) and iv) and hence has smaller variance. In either case, the variance in time measurements can have a significant impact on clock estimation and needs to be considered.

We will focus on solving this problem in the context of ad-hoc and wireless sensor networks. We propose a new clock synchronization algorithm called ACES, consisting of a novel clock model and Kalman filter to perform clock synchronization [16].

2.2 Existing Works

Many clock synchronization techniques for the Internet have been proposed over the past few decades, among which the most popular and widely used is Network Time Protocol (NTP). The development and evolution of NTP are described in Mills' classic papers [17, 18]. NTP uses NTP packets containing timestamp information exchanged between NTP server and the host across a network to perform time synchronization. NTP is designed to provide clock offset accuracy bounded by the round-trip time (RTT) between the server and the client.

However, NTP provides insufficient accuracy and robustness for many demanding applications. A few techniques have been proposed to improve measurement accuracy or clock stability. In [19] synchronization is offloaded onto a programmable network interface card. This card autonomously performs synchronization by sending periodic messages and performs timestamping when packets arrive. In [1, 20], a method of using a clock based on the more accurate CPU oscillator was proposed. This method relies on the high reliability of the processor oscillator and the availability of a Time Stamp Counter (TSC) register. The Precision Time Protocol [21] was drafted into the IEEE 1588 standard for synchronization of network measurement and control systems. It uses a specially designed network infrastructure to achieve high synchronization accuracy. Unfortunately, all these techniques may not be applicable to networks of low-cost devices.

Since some passive network monitoring tasks do not require real-time synchronization, a few algorithms have emerged for synchronizing data captures. In [22] a linear-programming based algorithm for estimating and removing the skew and offset of a data set was proposed. Convex hulls were used in [23] to estimate the clock skew and offset within a dataset. This was shown to perform better than the linear regression methods, but suffers increased computational complexity. All these algorithms are offline, which means they deal with the saved measurement data such as network packet delay traces instead of online clock synchronization.

In the Internet, each node is either a router or a host which is wired to a constant power source and has one or more stable and powerful CPUs. In contrast, some other networks have only very limited resources such as scarce energy, unstable processors, and unreliable low-bandwidth communications. WSN is a representative example of this type of networks. In a WSN, since the vast majority of sensors are battery-powered, a desirable clock synchronization scheme must preserve energy to prolong the battery life. Pottie et al. [24] shows that transmitting 1 bit over 100 meters requires 3 joules, which can be used for executing 3 million instructions. Therefore a successful clock synchronization scheme must minimize the amount of message exchange and at the same time maintain high synchronization accuracy. Scarcity of power on sensor nodes however is not the only resource constraint. Due to

its small size and low cost, the clock readings in a sensor are derived from oscillators with only limited stability (due to phase noise, thermal noise, aging, etc). Consequently, clocks on sensors are easily affected by temperature variations, vibration and interference and can significantly deviate from the reference sources [25, 26]. The situation could become even worse under catastrophic conditions such as earthquakes, battlefields, or forest fires. All these affect the clock drifting rates and make the clock drift nonlinear and time-varying.

WSNs bring a host of issues such as device quality and cost, many of which are not associated with wired networks. These issues lead to a number of new challenges to clock synchronization such as unstable oscillators, limited energy and communication bandwidth. Most of existing methods synchronize a sender with a receiver by transmitting the current clock values as timestamps [27–32]. In this regard, these methods are vulnerable to variance in message delay between the sender and the receiver due to network delays and processing overhead. Some other methods [33–36] perform receiver-to-receiver synchronization. These methods exploit the property of the physical broadcast medium where any receivers one-hop away receive the same message at approximately the same time. Such an approach reduces the message delay variance because non-deterministic delays at the transmitter no longer affect accuracy of the timestamp. Our proposed scheme is independent of the above synchronization modes. It can be easily adapted into both modes as long as we obtain reasonably good parameter estimates for the clock models.

Recent works have attempted to deal with the clock synchronization for WSNs from signal processing perspective. A survey in [37] categorizes the existing protocols and clock estimation results. In [38], assuming the delay model is known in two-way message exchanges, the maximum likelihood estimator (MLE) and its corresponding Cramér-Rao lower bound (CRLB) of clock offset and clock skew are derived. The authors proposed an ML-like skew estimator, which is simple and more suitable for WSNs. Later, the ML-like skew estimator in [38] was generalized in [39], where another estimator which reduces the complexity while bringing comparable performance to ML is proposed. These approaches estimate time-invariant clock skew and clock offset based on different delay models.

Kalman filtering has been used in the context of clock synchronization [40–42] for packet-switched networks. In [40], a Kalman filter was used to model the fluctuation in packet inter-arrival times, after shaping this fluctuation with low-pass prefiltering. In contrast, our work focuses on modeling a clock itself. A Kalman-filtering algorithm for end-to-end time synchronization has been presented [41]. This algorithm assumes a constant clock skew in the long term, which is not valid in most resource-constrained networks, and relies on NTP to exchange timestamp information. In [42], the author assumes constant clock skew and relies on the TSC register found in Pentium class CPUs, to count CPU clock cycles.

2.3 Clock Modeling

The time reported by a clock at some ideal time t is written as $C(t)$. We will write $C_A(t)$ as the time given by clock A at time t. The difference between the time of an ideal clock and a given clock is said to be the offset, $\theta(t)$, which is defined as

$$\theta(t) = C(t) - t.$$

The relative offset from node B to node A, $\theta_A^B(t)$, is defined as

$$\theta_A^B(t) = C_A(t) - C_B(t) = \theta_A(t) - \theta_B(t).$$

The oscillator in a clock produces periodic pulses. The difference between the rate these pulses are produced and the rate an ideal clock counts the desired interval is called the skew denoted by $\alpha(t)$:

$$\alpha(t) = \frac{d\theta(t)}{dt} \approx \frac{\theta(t+\tau) - \theta(t)}{\tau}. \tag{1}$$

The skew of a clock is the slope of the change in offset compared to the ideal clock. The slope of the relative offset $\theta_A^B(t)$ is relative skew $\alpha_A^B(t)$. This is defined as

$$\alpha_A^B(t) = \alpha_A(t) - \alpha_B(t).$$

If the oscillator were perfectly stable, the slope of $\theta(t)$ would reflect a constant skew α. However, this is not the case, especially in low-cost devices. Oscillators do not produce perfectly periodic pulses. The nonlinearity and the phase noise of the oscillator alter the

pulse period, making the clock rate time-varying [43]. Additionally, physical effects such as temperature and age can change the oscillator frequency. Without loss of generality, we will assume the reference node has a perfect clock (i.e., zero offset and skew) so that all the offset and skew notations lack subscripts. It is straightforward to adapt all of derivations into the relative sense when the reference node deviates from the actual time.

Here we borrow a figure from [1] to illustrate the randomness of clock offset and skew. The authors of [1] examined the clock offset of the same 600Mhz CPU host in two different temperature environments, *laboratory* which is an open plain area in a building without air conditioner, and *machine-room* which a closed temperature controlled room. In Figure 1 it is clear that the constant skew model fails over day timescale in both environments (the right figure), as the residual errors are far from linear. Recall that [1] adopts highly reliable CPU oscillators. We can expect that in a resource-constrained network low-cost quartz oscillators would generate more severe time-varying skews which are observable over smaller timescales (e.g., seconds or minutes).

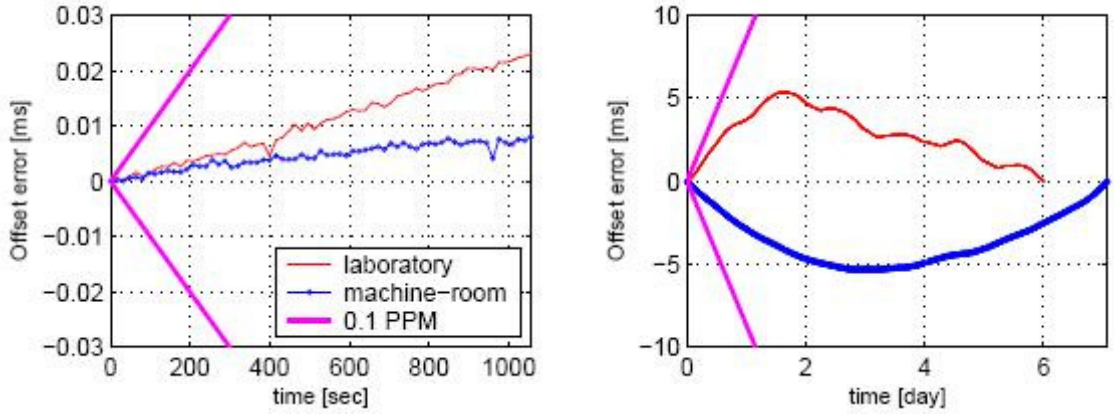


Figure 1: Illustration of clock drift (Figure 2 in [1]).

Having a complete understanding of clock drift, we decompose its variations into three independent components: the instantaneous clock skew $\alpha(t)$, the initial clock offset θ_0, and the random measurement and other types of additive noise $w(t)$. The instantaneous clock offset $\theta(t)$ at time t is given as:

$$\theta(t) = \int_0^t \alpha(\tau)\ d\tau + \theta_0 + w(t). \tag{2}$$

This model is quite general and subsumes all those existing simpler clock models. For example, if the clock skew $\alpha(t)$ does not change along with time t, the model in (2) reduces to the simple skew model in [1].

2.3.1 Discrete-Time Clock Model

After sampling, the continuous-time model becomes discrete-time model. In most cases a discrete clock model is desirable since the synchronization is typically achieved by timestamped message exchange. The timestamps are nothing but discrete samples of the continuous time. Based on (2), the discrete-time clock model is obtained as:

$$\theta[n] = \sum_{k=1}^{n} \alpha[k]\tau[k] + \theta_0 + w[n], \tag{3}$$

where k is the sample index, "$[\cdot]$" is adopted for discrete indexing, $\tau[k]$ is the sampling period at the k^{th} sample.

Here note that our discrete-time model is also quite general. It covers not only uniform sampling, but also non-uniform sampling (by choosing different $\tau[k]$). Since $w[n]$ is mainly caused by the observation and measurement noise, it is reasonable to assume $w[n]$'s are independently distributed with variance σ_w^2. The variance of $w[n]$ depends on the time-critical path [33].

We can rewrite this model using a recursive form as:

$$\theta[n] = \theta[n-1] + \alpha[n]\tau[n] + v[n], \tag{4}$$

where $v[n] = w[n] - w[n-1]$. Clearly, $v[n]$ is a random variable with mean 0 and variance $\sigma_v^2 = 2\sigma_w^2$. This is not surprising since (4) is the differential form of the observation equation for the clock, and differential forms are well-known to double the noise variance. Even if the observation noise $w[n]$ has non-zero mean, it is not difficult to verify that $v[n]$ still has zero mean because of the differential format in (4).

2.3.2 Clock-Skew Modeling

When using the recursive model in (4) to synchronize clocks, we need to estimate the clock skew $\alpha[n]$ which is also time-varying. Before we establish the clock skew model, we look at two extreme cases of clock skew.

Case i (constant skew): Suppose the clock skew $\alpha[n]$ is constant as in [1, 22]. From (4), since $\theta[k]$'s are known for $k = 1, \ldots, n$, if the sampling period $\tau[k]$'s are also known, the optimal clock skew estimator (in terms of mean-square error (MSE)) is

$$\hat{\alpha}[n] = \frac{\sum_{k=2}^{n}(\theta[k] - \theta[k-1])\tau[k]}{\sum_{k=2}^{n}\tau^2[k]}. \tag{5}$$

Case ii (independent skew): If the clock skew $\alpha[n]$ changes completely from one sample to another, the optimal estimator becomes

$$\hat{\alpha}[n] = \frac{\theta[n] - \theta[n-1]}{\tau[n]}. \tag{6}$$

These two cases are simple, but neither of them is practical. Most existing schemes are based on these two simple cases without considering any statistical and time-series models of the clock skew. Because of the phase noise in the oscillator, clock skew has certain randomness, but is not completely independent for each sample. Fig. 2 is an example of the real clock skew behaviors in a resource-constrained network. It shows the clock skews of the two clocks used in a low-powered micro-controller platform, which are 32.768kHz and 16MHz. We examined the clock skews using the platform in a temperature controlled room over 1.5 months. The variation of clock skew is observable over small timescales and it is clear that the constant skew model fails over timescales of several hours even in air controlled environments. It is expected that clock skew would vary severely in the real environment due to lack of energy, or large temperature variations. Therefore we investigate a model which can reflect these time-varying characteristics. In the following, we derive a model for the random clock skew starting from the phase noise.

Phase noise in oscillators has different representations. Here we consider a simple way to model it through jitter. It is natural to think of it as a noisy random offset in the timing of events. If the unperturbed oscillator output is $s(t)$, the jitter perturbed output is $s(t + \phi(t)/2\pi f_o)$, where $\phi(t)$ is random and f_o is the center frequency of the oscillator. Clearly the jitter $\phi(t)$ affects the frequency of the oscillator $\frac{d\phi(t)}{dt}\frac{1}{2\pi f_o}$ and thus causes clocks to have random offset and skew [26]. In general, phase noise is not stationary but only cyclostationary.

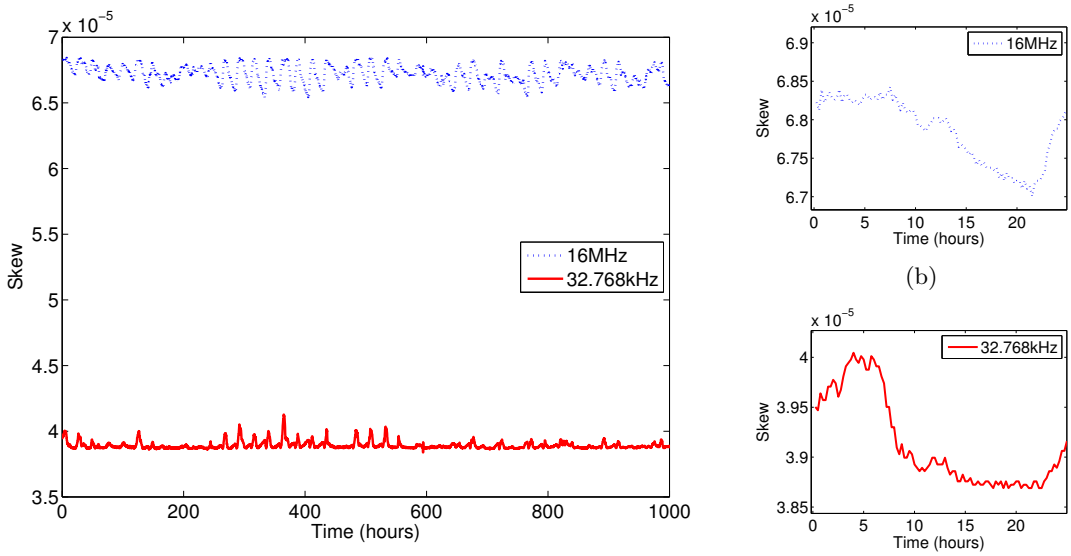


Figure 2: Measurement of time-varying clock skews

To model the time-varying clock skew as a random process, we assume clock skew is a random process with zero mean and a small perturbation around the mean. This assumption has been observed by some previous works (e.g., [1]) which adopt constant skew. We also assume the smoothness (order of autoregression model) of the clock skew is just first order due to the randomness of the phase noise. Therefore it is reasonable to adopt first-order Gauss-Markov model for the time-varying clock skew.

This means that the clock skew satisfies the auto-regressive (AR) relation as:

$$\alpha[n] = p\alpha[n-1] + \eta[n], \tag{7}$$

where p is a positive number less than but close to 1, $\eta[n]$ is model noise with zero mean and variance $\sigma_\eta^2 = (1-p^2)\sigma_\alpha^2$ with σ_α^2 being the variance of $\alpha[n]$.

Note that to obtain the variance, we only need to assume that $\alpha(t)$ is wide sense stationary. Usually we model $\eta[n]$ as Gaussian noise because in general, the phase noise derivative $\Delta\phi(t)$ is unbounded, but the frequency drift is focused within a certain range (which is usually specified by the oscillator manufacturer). This model is general and practical. It subsumes the two extreme cases as special cases. Although it is just first order, it quantifies the slow drifting of the clock frequency, captures the main variation of the skew and also

takes into account the randomness.

For this model to be useful, the parameter p needs to be determined. This parameter depends on the statistical properties of $\alpha[n]$. Define the auto-correlation function of $\alpha(t)$ as $r_\alpha(\tau) = E\{\alpha(t)\alpha(t+\tau)\}$, and then the AR(1) parameter p can be obtained as:

$$p = \frac{r_\alpha(\tau)}{r_\alpha(0)} = \frac{r_\alpha(\tau)}{\sigma_\alpha^2}. \tag{8}$$

It is clear that as time goes on, the auto-correlation of the clock skews becomes weaker. In [44], the auto-correlation function is modeled as a decaying exponential as:

$$r_\alpha(\tau) = \sigma_\alpha^2 \rho^{\frac{\tau}{\nu}},$$

where ν denotes the normalizer to model different decaying rates, and ρ is a positive number close to 1. We also adopt this exponential decaying model for the autocorrelation function. As τ increases, the auto-correlation $r_\alpha(\tau)$ decreases, and thus p reduces. Since the driven noise variance σ_η^2 also depends on p, when p is too small, the AR process is dominated by the noise and the tracking protocol may fail since the current value becomes too uncorrelated from the previous value.

To estimate the parameter p of AR(1) model, we need to estimate the auto-correlation function and variance of $\alpha[n]$. Given the clock observations θ_n in (1), one can obtain samples of the clock skew α_n's. Thus, the autocorrelation $r_\alpha(\tau_0)$ and the variance can be estimated using sample means. Once we obtain the auto-correlation, we can find its parameters by setting $\nu = \tau_0$ and solving for ρ. Then we can use this auto-correlation to estimate p for the desired sampling period τ. Theoretically $\alpha(t)$ is non-stationary, and thus parameter p may change along with time. However, p changes quite slowly relative to the clock offset and thus we can still take it as quasi-stationary.

If we take the model in (7) with parameters $\tau_0 = \nu$ and $p = \rho$, we can derive a statistically equivalent model using only every k^{th} sample. By only using every k^{th} sample, we are changing the effective sampling rate from ν to $k\nu$. This statistically equivalent model is

derived as follows:

$$\begin{aligned}
\alpha[n] &= \rho\alpha[n-1] + \eta[n] \\
&= \rho^2\alpha[n-2] + \rho\eta[n] + \eta[n] \\
&\vdots \\
\alpha[n] &= \rho^k\alpha[n-k] + \sum_{i=0}^{k-1} \rho^i\eta[n-i] \\
&= \rho^k\alpha[n-k] + \hat{\eta}[n] \qquad (9)
\end{aligned}$$

Note the similarity between (7) and (9). It is apparent that when $\tau_0 = k\nu$, $p = \rho^k = \rho^{\frac{k\nu}{\nu}}$. For the variance of the equivalent noise, $\hat{\eta}[n]$, we note that it is formed from the sum of Gaussian random variables with variances $\sigma_\eta^2, \rho^2\sigma_\eta^2, \rho^4\sigma_\eta^2, ..., \rho^{2(k-1)}\sigma_\eta^2$. Since, in this case, $\sigma_\eta^2 = \sigma_\alpha^2(1-\rho^2)$ we find $\mathrm{Var}[\hat{\eta}] = \sigma_\alpha^2(1-\rho^{2k}) = \sigma_\alpha^2(1-\rho^{\frac{2k\nu}{\nu}})$. If we extend this model to allow k to be real-valued, we can recalculate a model for any arbitrary sampling rate. If ρ and ν of a clock are known, p can be calculated as $p(\tau) = \rho^{\frac{\tau}{\nu}}$ for any desired sampling interval τ.

Given the recursive observation model in (4) and the AR model in (7), we are finally prepared to consider using Kalman filter as a framework to track the variation of the clock skew and synchronize the clocks. In the following section, we will introduce the ACES algorithm for time synchronization.

2.4 Tracking Clock Skew and Offset

Ideally, clock behavior could be estimated accurately if an exact model is derived. In the real world, however, clocks are affected by environmental factors such as temperature variations, vibration and humidity. Moreover, resource-constrained networks, such as WSNs usually consist of inexpensive devices which have unstable oscillators, vulnerable to interference. These factors have a nondeterministic effects on clock behavior, which prevents even an exact clock model from tracking clock behaviors exactly. In this section, we design Kalman filters to track the time-varying clock skew and offset based on the proposed clock skew and clock offset models.

2.4.1 Kalman Filter

Let $\tilde{\theta}[n]$ denote the true clock offset (i.e., $\tilde{\theta}[n] = \sum_{k=1}^{n} \alpha[k]\tau[k] + \theta_0$). It is clear that $\tilde{\theta}[n] = \tilde{\theta}[n-1] + \alpha[n]\tau_0$. Based on the AR(1) model in (7), we can define an extended state equation as

$$\boldsymbol{x}[n] = \boldsymbol{A}\boldsymbol{x}[n-1] + \boldsymbol{u}[n], \tag{10}$$

where $\boldsymbol{x}[n] = \begin{bmatrix} \tilde{\theta}[n] & \alpha[n] \end{bmatrix}^T$, $\boldsymbol{A} = \begin{bmatrix} 1 & \tau_0 \\ 0 & p \end{bmatrix}$, and $\boldsymbol{u}[n] = \begin{bmatrix} 0 & \eta[n] \end{bmatrix}^T$. The observation equation is the noisy observation of the offset:

$$\theta[n] = \tilde{\theta}[n] + v[n] = \boldsymbol{b}^T\boldsymbol{x}[n] + v[n], \tag{11}$$

where $\boldsymbol{b}^T = \begin{bmatrix} 1 & 0 \end{bmatrix}$. In this case, the Kalman filter design is summarized as follows.

$$\text{Update}: \hat{\boldsymbol{x}[n]} = \boldsymbol{A}\hat{\boldsymbol{x}}[n-1] + \boldsymbol{G}[n](\theta[n] - \boldsymbol{b}^T\boldsymbol{A}\hat{\boldsymbol{x}}[n-1]) \tag{12}$$

$$\text{MSE}: \boldsymbol{\Sigma}[n] = \boldsymbol{A}\boldsymbol{M}[n-1]\boldsymbol{A}^T + \boldsymbol{C}_u \tag{13}$$

$$\boldsymbol{M}[n] = (\boldsymbol{I} - \boldsymbol{G}[n]\boldsymbol{b}^T)\boldsymbol{\Sigma}[n] \tag{14}$$

$$\text{Kalman Gain}: \boldsymbol{G}[n] = \boldsymbol{\Sigma}[n]\boldsymbol{b}\left(\sigma_v^2 + \boldsymbol{b}^T\boldsymbol{\Sigma}[n]\boldsymbol{b}\right)^{-1}, \tag{15}$$

where $\boldsymbol{\Sigma}[n]$ is the prediction MSE of the estimate when the observation is not considered, $\boldsymbol{A}\hat{\boldsymbol{x}}[n-1]$. The recursion of the Kalman filter is initialized by

$$\hat{\boldsymbol{x}}[0] = \begin{bmatrix} \mathrm{E}\{\theta[n]\} \\ \mathrm{E}\{\alpha[n]\} \end{bmatrix}, \quad \boldsymbol{M}[0] = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\alpha^2 \end{bmatrix}$$

An example profile of this algorithm is shown in Figure 3. The observation noise variance is $\sigma_v^2 = 10^{-5}$ s^2, and the parameter ρ is chosen as $1 - 2 \cdot 10^{-6}$ with $\nu = 1$ hour. Figure 3 shows the offset estimated by our algorithm from (12) - (15) ("Estimated $\theta[n]$"), the true offset $\tilde{\theta}[n]$ ("True $\theta[n]$"), and the offset from observation ("Observed $\theta[n]$"). Note that even though the observed offset is completely dominated by the observation noise, the Kalman filter is able to extract the true value with only small deviations.

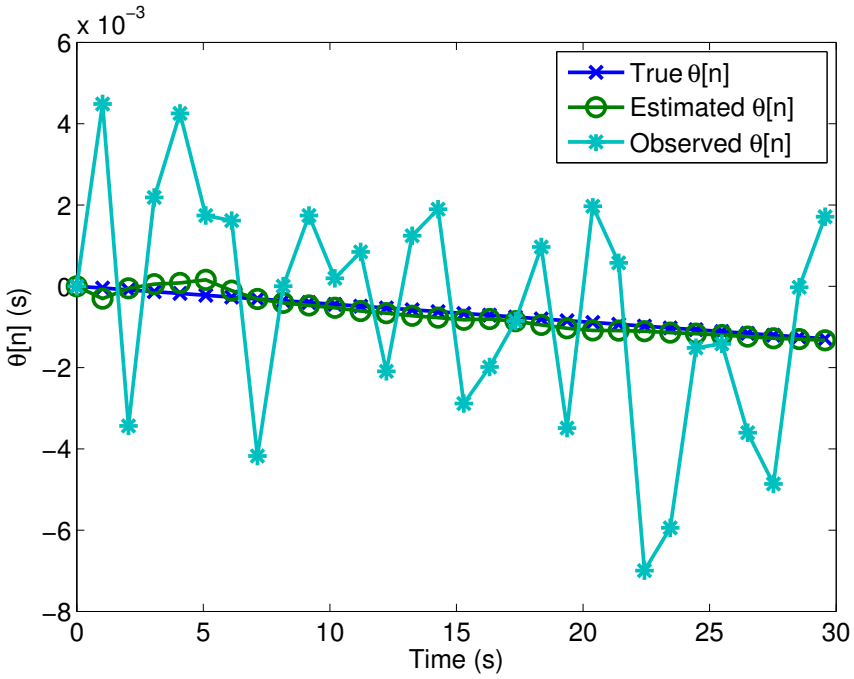
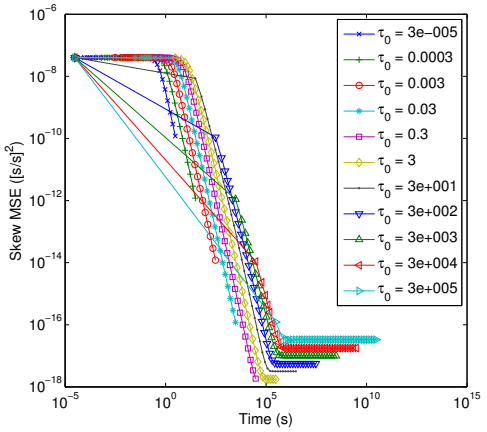


Figure 3: Clock offset tracking example

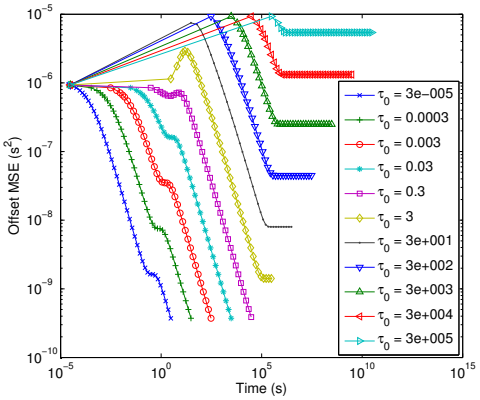
2.4.2 MSE Analysis

The convergence speed of the vector Kalman filter method is greatly affected by the synchronization period chosen. This feature is analyzed through simulation. The tracking algorithm is run for 100,000 synchronization periods with several different synchronization period lengths. The results are shown in Figure 4. The skew MSE is shown in Figure 4(a) and the offset MSE is shown in Figure 4(b). skew MSE converges in fewer synchronization periods if the synchronization periods are longer. Figure 4(a) shows that this is still true for the vector Kalman filter. This can also be seen to be true for the offset MSE as well. In spite of this rapid (in terms of synchronization periods) convergence, the longer synchronization periods result in a longer time before the nodes are synchronized. Additionally, the longer synchronization periods offer worse steady-state offset MSE.

This motivates us to investigate the steady-state prediction MSE for this model. Unfortunately, no closed form solution could be found. Instead, we run a simulation to estimate the final steady-state MSE. Figure 5 shows the effects of sampling frequency and measurement noise. In Figure 5(a) the MSE of the clock skew estimate is shown. If we consider the

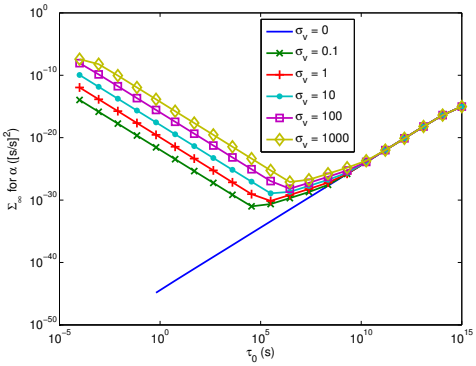


(a) Skew MSE vs. time for various τ_0

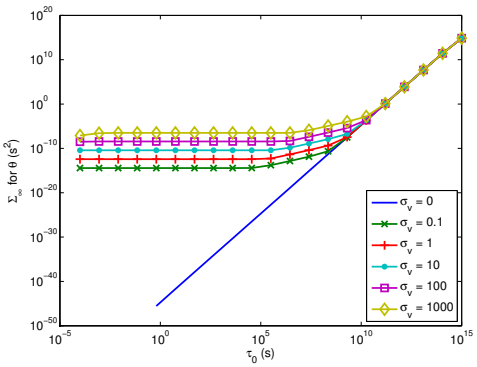


(b) Offset MSE vs. time for various τ_0

Figure 4: Simulated MSEs vs. time



(a) Σ_∞ for α using offset tracking model



(b) Σ_∞ for θ using offset tracking model

Figure 5: Simulated $\mathbf{\Sigma}_\infty$ for offset tracking

effect of the reduced MSE in the estimate of α with increasing τ_0, we would expect this to cause the offset estimate's MSE to decrease. Note from Figure 5(b) that this decrease in MSE for α does not appear to significantly affect the offset estimate.

Table 1 provides some numerical results on the steady-state prediction MSEs for several different sampling periods and oscillator characteristics. The observation noise variance is $\sigma_v = 10^{-5}$ s^2, and the parameter ρ is chosen as $1 - 2 \cdot 10^{-6}$ with $\nu = 1$ hour. This table shows the high performance of our method. As the oscillator accuracy increases, the tracking performance is also improved. It can also be used to determine the clock synchronization period required for a desired offset tracking accuracy for a system that uses oscillators with a certain characteristic.

Table 1: Skew and offset estimation accuracy

Sampling Period τ_0 (s)	Oscillator Instability σ_α (ppm)	Steady-state MSE of θ (s^2)	Steady-state MSE of α $([s/s]^2)$
1	2	1.90×10^{-15}	4.25×10^{-23}
60	2	4.21×10^{-14}	1.19×10^{-22}
900	2	3.25×10^{-13}	2.37×10^{-22}
1800	2	5.54×10^{-13}	2.85×10^{-22}
3600	2	9.50×10^{-13}	3.46×10^{-22}
1	20	6.00×10^{-15}	1.34×10^{-21}
60	20	1.34×10^{-13}	3.77×10^{-21}
900	20	1.07×10^{-12}	7.77×10^{-21}
1800	20	1.87×10^{-12}	9.58×10^{-21}
3600	20	3.36×10^{-12}	1.21×10^{-20}
1	200	2.00×10^{-14}	4.25×10^{-20}
60	200	4.30×10^{-13}	1.21×10^{-19}
900	200	3.83×10^{-12}	2.75×10^{-19}
1800	200	7.35×10^{-12}	3.63×10^{-19}
3600	200	1.57×10^{-11}	5.09×10^{-20}

2.5 Concluding Remarks

Efficient and accurate network time synchronization is a challenging problem in resource-constrained networks. We have described a novel Kalman filter based approach to achieve high efficiency in terms of both energy and communication bandwidth. Considering the inherent instability of the inexpensive oscillators, we proposed general models to capture the time-varying behavior of clock offset and skew. Then, applying these models we generated a clock skew and offset estimation algorithm based on the Kalman filter. We analyzed the performance of this algorithm in detail for both the time-varying and steady-state cases.

This particular work has since been continued by Hayang Kim. In her later investigations [45, 46], we collaborated to take measurements from low quality clocks and then she developed a more accurate clock model and synchronization algorithm.

CHAPTER III

RF TOMOGRAPHY

In dense networks with large numbers of wireless devices, the received signal strength (RSS) measurements between all the wireless nodes in the network contain a significant amount of information about the physical environment the network is operating in. As radio transmissions pass through the environment, the signal is attenuated by the objects it passes through, causing shadowing. With an appropriate shadowing model and processing, this previously unavailable information can be used in a variety of applications.

Traditional shadowing models such as the log-normal shadowing model [47] assume shadowing for each link is independent, in reality the shadowing present in links among adjacent nodes is likely to be correlated. Other shadowing models [48, 49]) have described the shadowing correlation when one of the nodes in the link is mobile, their models do not generalize to to model the shadowing in the entire network, so they are unable to model the environmental information from RSS measurements.

Recent works [2,50–53] have instead represented the shadowing component as a function of an underlying spatial loss field $g(\boldsymbol{s})$. This spatial loss field, represents the additional attenuation due to shadowing at each point in space. The shadowing component of a link can then be found as some linear function of the spatial loss field. Estimating the spatial loss field can then be shown to be a tomography problem.

Tomography is the method of imaging a planar section of an object, through the use of a penetrating wave, such as the use of x-rays in medical imaging computed axial tomography (CAT) scans, sometimes referred to as computed tomography (CT) scans. In CT scans, a circular array of x-ray emitters and receivers are positioned around the object of interest. The total path loss between the transmitters and receivers can be used to determine the spatial loss field. This spatial loss field represents how much each point in space contributes to the total path loss experienced across all of the measured paths, and is effectively a

cross-sectional image of the object. If there are sufficient measurements following unique paths in the network, tomography techniques such as filtered backprojection [54] can be used to construct an image of the attenuation in the network. This technique, called Radio Frequency (RF) Tomography is a way to image passive objects using only received signal strength (RSS) measurements.

RF tomography has the ability to construct images of the network area, even when there are obstructions or other factors that would limit visibility. For this reason RF tomography can be useful in situations from locating survivors in rescue operations to providing comprehensive intelligence for tactical assaults. Further, when combined with an appropriate shadowing model, RF tomography can be used to improve channel estimates between nodes. The improved channel estimates are useful for site surveys since it reduces the number of samples required to achieve a certain accuracy. Additionally, as we will discuss in Chapter 4, such channel estimates can be used to improve the accuracy of wireless localization. RF tomography can also be used to track movement and location of people or objects through "device-free passive localization" [55].

3.1 Problem Statement

RF tomography has many similarities to the traditional tomographic reconstruction problem. In both X-ray and RF tomography, power of the transmitted wave suffers attenuation from objects that the wave passes through. The tomographic reconstruction problem is to estimate and image of the attenuating objects in the environment from a series of measurements. For the case of higher frequencies, such as X-rays, the total signal attenuation corresponds to a multiplicative attenuation at each point in space such that the ratio of received power to transmitted power for a signal traveling a line-of-sight path $(x(t), y(t))$, $0 \leq t \leq T$ is

$$P_r/P_t = 10^{\frac{\int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} g(\boldsymbol{s})\ d\boldsymbol{s}}{10}} d_{i,j}^{-\alpha}, \tag{16}$$

where $d_{i,j}$ is the path distance and $g(\boldsymbol{s})$ is the spatial loss field describing the attenuation at each point. If the free-space pathloss is removed, the additional loss due to shadowing

can be expressed in dB as

$$\eta = \int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} g(\boldsymbol{s})\ d\boldsymbol{s}. \tag{17}$$

For practical purposes, the spatial loss field is always discretized so that instead of representing how every point in space contributes to the total path loss, it aggregates the points into pixels (or voxels) and describes how much each pixel contributes to the total path loss. In the discretized version of the model, the field is replaced by a column vector $\boldsymbol{g}$. The exact relationship between the continuous field $g(\boldsymbol{s})$ and the discrete approximation depends on the model. The line integral in (17) is then replaced by a vector product such that:

$$\eta = \boldsymbol{b}\boldsymbol{g}, \tag{18}$$

where $\boldsymbol{b}$ is the discrete tomographic projection vector with elements corresponding to each discrete (x,y) pixel in the field $\boldsymbol{g}$.

For example, consider Figure 6. In Figure 6(a) we have several transceivers (red circles) which take path loss measurements across each of the (black) paths to the other transceivers. Since the measurement (in dB) between any given pair of transceivers A and B are linear functions of the spatial loss field such that $\eta_{AB} = \boldsymbol{b}_{AB}\boldsymbol{g}$, we can rewrite this in matrix form as

$$\boldsymbol{\eta} = \boldsymbol{B}\boldsymbol{g}, \tag{19}$$

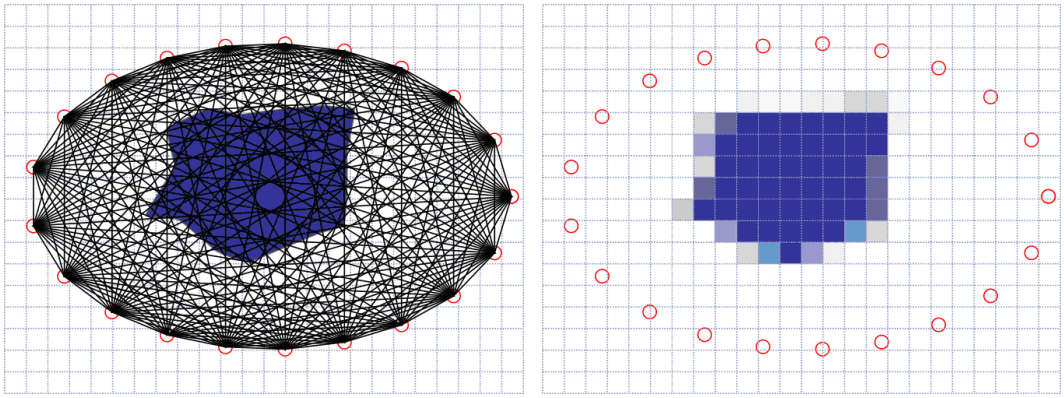
where $\boldsymbol{B}$ is a matrix containing the row vectors $\boldsymbol{b}_{AB}$ for all transceivers pairs (A, B) and the vector $\boldsymbol{\eta}$ contains the corresponding measurements, η_{AB}. Then the spatial loss field can be estimated as

$$\hat{\boldsymbol{g}} = \boldsymbol{B}^{-1}\boldsymbol{\eta}. \tag{20}$$

In most cases, however this inversion is not possible since B is under-determined. To compensate for this, regularization techniques such as Tikhonov regularization are applied, resulting in the reconstruction

$$\hat{\boldsymbol{g}} = \left(\boldsymbol{B}^H\boldsymbol{B} + \boldsymbol{C}_g^{-1}\right)^{-1}\boldsymbol{B}^H\boldsymbol{\eta}. \tag{21}$$

A visual example of the tomographic reconstruction corresponding to the measurements in Figure 6(a) can be found in Figure 6(b).



(a) Signal strength measurements of shadowing from an object

(b) Tomographic reconstruction of the object from the measurements

Figure 6: Shadowing example

Unfortunately, while RF tomography is both cheaper and safer than X-ray tomography, it faces additional challenges. The lower frequency RF signals are subject to scattering, diffraction, and other non-line-of-sight (NLOS) propagation in common network environments. The NLOS components in the received signal may be significant, so the simple linear weighted tomographic projection model described above will not accurately model the actual operating environment. The NLOS components in the received signal may be significant, so the simple linear weighted tomographic projection model described above will not accurately model the actual operating environment. Additionally the wireless devices making the measurements in the network may be stationed at arbitrary positions or even be mobile.

Accurate tomographic reconstructions generally require measurements that follow a substantial number of unique paths through the imaging area. If the wireless nodes are not mobile, this means that a very large density is required to have the requisite number of unique paths. If on the other hand, the nodes are considered mobile, practically every measurement will follow a unique path through the network. Relying on node mobility spreads the measurements out over time, however, and presents a significant challenge for motion detection and tracking since RSS measurements from different time frames can no longer be directly compared.

We consider three aspects of the RF tomography problem. In Section 3.4, we examine

the problem of using RF tomography to both image a stationary structure and detecting and tracking motion. We propose a method using a Kalman filter to jointly estimate an image of the stationary structure and and image of moving objects, and demonstrate that it offers better performance than existing methods through simulation.

In Section 3.5, we describe our RF tomography testbed and compare existing RF propagation models for RF tomography with our novel model. We use measurements from a field test to examine how consistent each of the models is with the actual environment from the field test.

3.2 Existing Works

Although tomographic techniques have been used with RF frequencies for imaging in geophysics for some time [56,57], these methods consider the more complicated situation where scattering and reflective effects are significant. To compensate for these effects they require the phase information from the received signal. Then they combine techniques from both tomography and ground penetrating radar to image tunnels under ground.

Several recent works have adopted linear models to approximate the effects of shadowing to make the problem more tractable. Such works have adopted a shadowing model where the shadowing component of the wireless links as a function $b(g)$ of an underlying spatial loss field $g(\boldsymbol{s})$ which represents the additional attenuation due to shadowing at each point $\boldsymbol{s}$ in space. In [53], the function $b(g)$ was assumed to be simply the line integral of the spatial loss field from the source to the destination

$$\int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} g(\boldsymbol{s})\ d\boldsymbol{s}.$$

This model was not verified experimentally, so it may not correspond to shadowing effects experienced in realistic environments. The Network Shadowing (NeSh) model [51], also uses a line integral as the function $b(g)$, but applies an additional weight based on the distance from the source to destination

$$\int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} \frac{1}{\sqrt{d}} g(\boldsymbol{s})\ d\boldsymbol{s}.$$

This model was developed based on both a set experimental of measurements and correspondence with large scale shadowing models such as the log-normal shadowing model.

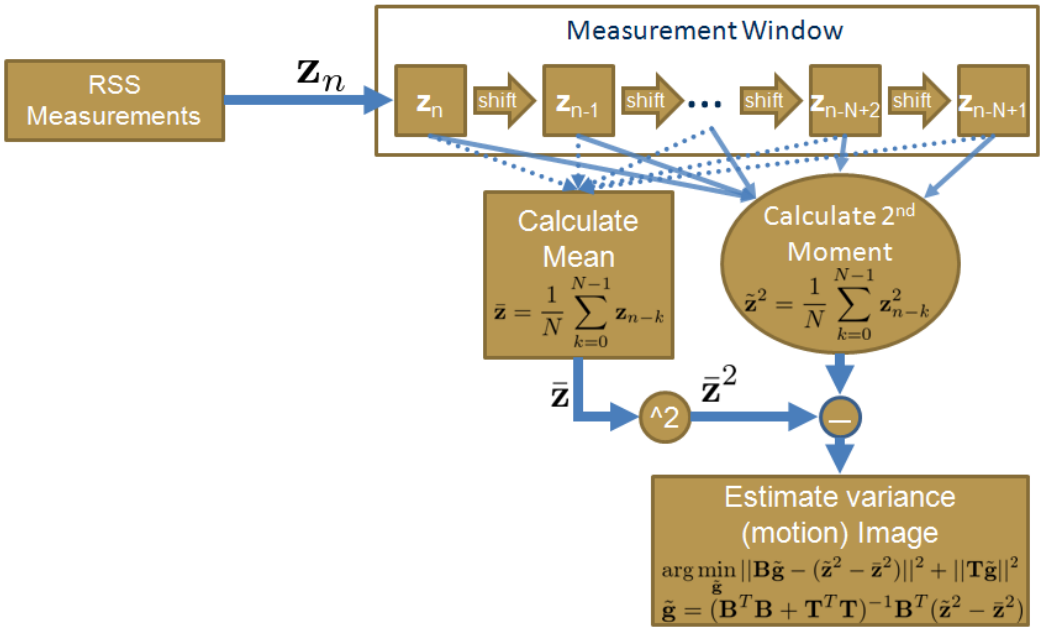


Figure 7: Block diagram of the algorithm in [2]

A few other works [2, 52] have used area integrals of an ellipse with the transmitter and receiver as foci, which was justified as an approximation of the NeSh model. While the NeSh model was based on experimental measurements, only shadowing was considered. No previous works have investigated how well this model works for RF tomographic imaging.

Wilson et al [2, 52] also investigated methods of detecting motion or changes in the environment with RF tomography. Their method works by obtaining the windowed variance of the RSS measurements, and projecting this variance back into the image domain. A block diagram is shown in Figure 7. While this method's ingenious use of statistics on the RSS measurements allows it to see through the noise and artifacts from modeling error it has several limitations. The reliance on RSS measurement statistics means that it cannot be applied in networks where the nodes making the measurements are mobile. Further, the use of the windowed variance means that reconstructed images of the moving objects consist of motion blurs. These blurs begin weakly from the past position at the beginning of the window, reach a maximum value at the middle of the window, and then fade as they approach the moving object's current position. This makes this algorithm less useful for real-time intelligence.

Although this algorithm was not designed to estimate stationary shadowing components in the network, we will use a small modification which estimates the stationary shadowing based on the windowed mean RSS measurements to facilitate comparison.

3.3 RF Channel Model

We assume a single-path channel between two nodes with both shadowing and path loss. This model can be extended to multi-path channels if only the first arriving path is considered, and the other delayed paths treated as noise. The transmitted signal x is received as:

$$y = hx + u,$$

where h is the wireless channel and u is additive white Gaussian noise (AWGN). The channel h is

$$h = 10^{\frac{\eta}{20}} d^{-\alpha},$$

where d is the distance between transmitter and receiver, α is the path loss exponent (nominally between 2 and 4), and η is the shadowing component.

The shadowing component is some function of the spatial loss field (SLF) $g(\boldsymbol{s})$, which is assumed Gaussian distributed, with a covariance between points separated by a distance $d_{i,j}$:

$$R_g(d_{i,j}) = \frac{\sigma_\eta^2}{\kappa} e^{-\frac{d_{i,j}}{\kappa}}, \tag{22}$$

where σ_η^2 is the shadowing covariance and κ is a parameter controlling how fast the correlation falls off with distance.

We assume the RSS measurements are proportional to the channel estimate. In practical systems the proportionality constant can be determined and eliminated, so without loss of generality we can model the RSS measurement $z_{i,j}$ from node i to node j as:

$$z_{i,j} = h_{i,j} + v_{i,j}, \tag{23}$$

where $v_{i,j}$ is AWGN with variance σ_v^2, and the channel $h_{i,j}$ is

$$h_{i,j} = 10^{\frac{\eta_{i,j}}{20}} d_{i,j}^{-\alpha}. \tag{24}$$

Due to the NLOS effects, an accurate calculation of the shadowing component from the spatial loss field in the networks area is nonlinear. Such non-linear tomographic projection models are difficult to deal with in practice, so we consider three linear tomographic projection models. More specifically, we examine shadowing models such that the shadowing on the link between nodes at positions $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ is found as the spatial integral of the spatial loss field over the network area weighted by some function $b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$:

$$\eta_{i,j} = \int g(\boldsymbol{s}) b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) \quad d\boldsymbol{s}. \tag{25}$$

These weighting functions are then adapted into weight vectors for the discretized spatial loss field. A visual comparison of the weight vectors for the three different shadowing models we are considering is shown in Figure 8, and they are each described in more detail below.

NeSh Model

In a line of sight tomographic model, the path loss from shadowing is assumed to be proportional to a line integral across the spatial loss field. This is the most common model used in CT, and a modified version of this model for radio frequencies was introduced as the Network Shadowing Model (NeSH) [50,51]. The NeSH model differs from traditional tomographic line integral model in that the weights are multiplied by the square root of the path length. This means the tomographic projection is

$$b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) = \frac{1}{\sqrt{d_{i,j}}} \int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} \delta(|\boldsymbol{s} - \tilde{\boldsymbol{s}}|) \quad d\tilde{\boldsymbol{s}},$$

where $\delta(x)$ is the Dirac delta function. With this model, the shadowing component simplifies as

$$\eta_{i,j} = \frac{1}{\sqrt{d_{i,j}}} \int_{\boldsymbol{s}_i}^{\boldsymbol{s}_j} g(\boldsymbol{s}) \quad d\boldsymbol{s}. \tag{26}$$

This modification was made so that the model matches the larger scale shadowing statistical models. It was necessary because this model does not consider the effects of diffraction at all. This modification only changes the large scale statistics. This means that, even with this modification, diffraction will cause significant artifacts in the reconstruction.

As mentioned in Section 3.1, a discretized tomographic projection vector $\boldsymbol{b}$ is usually used instead of the continuous tomographic projection function $b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ listed above. The tomographic projection vector $\boldsymbol{b}_{i,j}$ for this model is a vector of weights with entries equal to length of the path that falls within the corresponding pixel, divided by the square root of the path length. A visual example of the weights from this model is shown in Figure 8(a).

Normalized Ellipse Model

In this model, the tomographic projection consists of a selection function based on an ellipse that has the transmitter and receiver as foci and some semi-minor axis length λ. The selection function has a value of $\frac{1}{\sqrt{d_{i,j}}}$ inside the ellipse, and 0 elsewhere. This model has also been previously used for RF Tomography [2, 52, 58]. Mathematically we can represent this model as

$$b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) = \begin{cases} \frac{1}{\sqrt{d_{i,j}}} & \frac{p_x(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})^2}{d_{ij}^2 + \lambda^2} + \frac{p_y(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})^2}{\lambda^2} < 1 \\ 0 & \text{else} \end{cases},$$

where $p_x(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ and $p_y(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ are projection functions that project the point $\boldsymbol{s}$ onto an axis aligned parallel to the line from $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, and centered between $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$.

Based on geometry, we can define the projections functions $p_x(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ and $p_y(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ as

$$p_x(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) = \boldsymbol{\psi}_x(\boldsymbol{s}_x - \boldsymbol{c}_x) + \boldsymbol{\psi}_y(\boldsymbol{s}_y - \boldsymbol{c}_y)$$

$$p_y(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) = -\boldsymbol{\psi}_y(\boldsymbol{s}_x - \boldsymbol{c}_x) + \boldsymbol{\psi}_x(\boldsymbol{s}_y - \boldsymbol{c}_y),$$

where the projection vector $\boldsymbol{\psi}$ is

$$\boldsymbol{\psi} = \frac{\boldsymbol{s}_i - \boldsymbol{s}_j}{||\boldsymbol{s}_i - \boldsymbol{s}_j||}$$

and $\boldsymbol{c}$ is the position of the center of the ellipse

$$\boldsymbol{c} = \frac{\boldsymbol{s}_i + \boldsymbol{s}_j}{2}.$$

There is some physical justification for using an ellipse, as the well-known Fresnel zone has an ellipsoidal shape. A problem with this model is that there is not a good physical basis for determining the parameter λ. Current methods using this model find the parameters of the ellipse through trial and error. Additionally the decision to set the weights of all of the pixels equally also has no physical justification.

The discrete tomographic projection vector $\boldsymbol{b}_{i,j}$ corresponding to this model has weights for each pixel are equal to the continuous projection function $b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$, evaluated at the center of the pixel. For an NxN field $\boldsymbol{g}$ with pixels centered at positions $\boldsymbol{s}_{p_{x,y}}$, we can write

$$\begin{aligned}\boldsymbol{b}_{i,j} = \big[&b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{0,0}}) \ , \ b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{1,0}}), \ \cdots , \\ &b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{N-1,0}}), \ b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{0,1}}), \ b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{1,1}}), \ \cdots , \ b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}_{p_{N-1,N-1}})\big] \quad (27)\end{aligned}$$

A visual example of this model is shown in Figure 8(b).

Inverse Area Elliptical Model

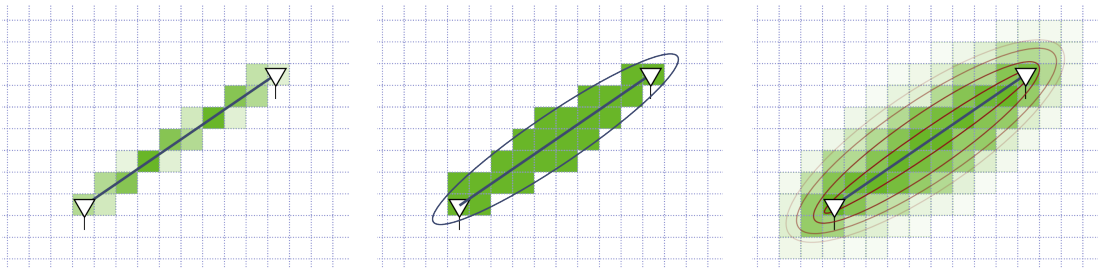
We introduce the Inverse Area Elliptical Model. In this model, the tomographic weighting function $b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$ is equal to the inverse of the area of the smallest ellipse that has the transmitter and receiver as foci. Mathematically we can represent this model as

$$b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s}) = \frac{1}{\pi\tilde{\lambda}\sqrt{d_{i,j}^2 + \tilde{\lambda}^2}},$$

where the parameter $\tilde{\lambda}$ is the length of the semi-minor axis of the smallest ellipse containing the point $\boldsymbol{s}$. More formally,

$$\tilde{\lambda} = \arg\min_{\lambda} \left| \frac{p_x(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})^2}{d_{ij}^2 + \lambda^2} + \frac{p_y(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})^2}{\lambda^2} - 1 \right|.$$

We bound the weights by setting minimum and maximum semi-minor axis lengths $\lambda_{\min}$ and $\lambda_{\max}$. The weight for points within the smallest ellipse is equal to the inverse of the area of the smallest ellipse. The weight for points outside the largest ellipse are set to 0. We set $\lambda_{\max}$ to be the semi-minor axis length of the first Fresnel ellipse. This model is more complicated than the Normalized Ellipse model, but should better



(a) Line Integral Model (b) Normalized Ellipse Model (c) Inverse Area Elliptical Model

Figure 8: Example weight vectors for the three tomographic models

capture the relative strengths of diffracted paths: paths that travel further than the line of sight path will be weaker from traveling the extra distance, so they will have a smaller contribution to the total RSS.

For the discrete tomographic projection vector based on this model, the weights for each pixel are equal to the continuous weight $b(\boldsymbol{s}_i, \boldsymbol{s}_j, \boldsymbol{s})$, evaluated at the center of the pixel. This is analogous to the conversion described in Eq. (27) for the Normalized Ellipse model. A visual example of this model is shown in Figure 8(c).

3.4 RF Tomography with Mobile Nodes

Current RF tomography algorithms do not work when the sensing nodes are mobile. We present the RF Exploitation for Tomographic Imaging and Non-cooperative Analysis (RETINA) algorithm for RF tomography with mobile nodes [59]. RETINA uses a recursive least-squares approach to perform real-time imaging of both the static and dynamic shadowing components in the network area. It uses a motion model to accurately separate and track moving objects in the network area. Because it performs the estimation in the image domain, instead of on the individual measurements, it is able to work when nodes in the network are mobile. Further, the use of the motion model allows for better time resolution of the dynamic shadowing components in the network.

3.4.1 The RETINA algorithm

We represent the state of the spatial loss field at time n as a vector $\boldsymbol{\theta}[n]$ containing both the static and dynamic components: $\boldsymbol{\theta}[n] = \begin{bmatrix} \boldsymbol{g}^T & \tilde{\boldsymbol{g}}[n]^T \end{bmatrix}^T$. At each measurement interval

n, the shadowing measurement $\boldsymbol{z}_n$ is

$$\boldsymbol{z}_n = \begin{bmatrix} \boldsymbol{B}[n] & \boldsymbol{B}[n] \end{bmatrix} \boldsymbol{\theta}[n], \tag{28}$$

where $\boldsymbol{B}[n]$ is the matrix of weight vectors determined according to the shadowing model.

Moving objects in the network area are assumed to move so that the difference between their next and previous position is Gaussian. This is effectively an auto-regressive (AR) position model since the future position of objects is related to the current position $\boldsymbol{s}[n]$ position as

$$\boldsymbol{s}[n+1] = \boldsymbol{s}[n] + \boldsymbol{v}[n], \tag{29}$$

where $\boldsymbol{v}[n]$ is iid Gaussian random vector with mean $\bar{\boldsymbol{v}} = \boldsymbol{0}$ and covariance $\boldsymbol{C}_v = \sigma_v^2 \boldsymbol{I}$. This model cannot be directly used, however, since RETINA does not track objects, but instead tracks changes in the spatial loss field induced by such objects.

We can determine the effect on the dynamic image by considering what would happen to a cloud of point-like shadowing objects obeying this AR model. If each point in the cloud has magnitude a_k and position $\boldsymbol{s}_k$, we can write the dynamic continuous spatial loss field from the cloud as a function of position $\boldsymbol{s}$:

$$\tilde{g}(\boldsymbol{s})[n] = \sum_k a_k \delta\left(||\boldsymbol{s} - \boldsymbol{s}_k[n]||\right), \tag{30}$$

where $\delta(x)$ is the Dirac delta function. Since the points in the cloud all follow (29), the continuous spatial loss field at the next time frame can be written as

$$\tilde{g}(\boldsymbol{s})[n+1] = \sum_k a_k \delta\left(||\boldsymbol{s} - \boldsymbol{s}_k[n] + \boldsymbol{v}_k[n]||\right). \tag{31}$$

If we take the expected value of (31), we find that the average effect of this movement model is to apply a Gaussian smoothing filter (with covariance $\boldsymbol{C}_v$) to the dynamic spatial loss field. Returning to the discrete field $\tilde{\boldsymbol{g}}[n]$, we can write the analogous state equation for dynamic component of the expected spatial loss field as

$$\tilde{\boldsymbol{g}}[n+1] = \boldsymbol{\Gamma}\tilde{\boldsymbol{g}}[n], \tag{32}$$

where $\boldsymbol{\Gamma}$ is the 2-D Gaussian filter operator. We then can find the covariance of this spatial loss field estimate as

$$\boldsymbol{C}_{\tilde{g}}[n] = \mathrm{diag}(\boldsymbol{\Gamma}\tilde{\boldsymbol{g}}[n]^2) - \boldsymbol{\Gamma}\tilde{\boldsymbol{g}}[n]\tilde{\boldsymbol{g}}[n]^T\boldsymbol{\Gamma}^T, \tag{33}$$

where 'diag($\boldsymbol{x}$)' denotes diagonal matrix containing the vector $\boldsymbol{x}$ on its diagonal.

Given these state equations, we formulate RETINA based on the well-known Kalman filter [5]. We represent the spatial loss field estimate at time n based on the first k measurements as $\hat{\boldsymbol{\theta}}[n|k]$, and its variance as $\boldsymbol{P}[n|k]$. The Kalman filter consists of two main parts: the measurement update and the state update.

In the measurement update, the current set of measurements $\hat{\boldsymbol{z}}[n]$ is incorporated into the current spatial loss field estimate. For simplicity we define $\boldsymbol{H}[n] = \begin{bmatrix} \boldsymbol{B}[n] & \boldsymbol{B}[n] \end{bmatrix}$ and $\boldsymbol{C}_u = \sigma_u^2 \boldsymbol{I}$.

$$\boldsymbol{S}[n] = \boldsymbol{H}[n]\boldsymbol{P}[n|n-1]\boldsymbol{H}[n]^H + \boldsymbol{C}_u \tag{34}$$

$$\boldsymbol{K}[n] = \boldsymbol{P}[n|n-1]\boldsymbol{H}[n]^H \boldsymbol{S}[n]^{-1} \tag{35}$$

$$\hat{\boldsymbol{\theta}}[n|n] = \hat{\boldsymbol{\theta}}[n|n-1] + \boldsymbol{K}[n]\left(\hat{\boldsymbol{z}}[n] - \boldsymbol{H}[n]\hat{\boldsymbol{\theta}}[n|n-1]\right) \tag{36}$$

$$\boldsymbol{P}[n|n] = \left(\boldsymbol{I} - \boldsymbol{P}[n|n-1]\boldsymbol{H}[n]\right)\boldsymbol{P}[n|n-1]. \tag{37}$$

In the state update, the state update matrix $\boldsymbol{F}$ is applied to the previous state $\hat{\boldsymbol{\theta}}[n-1|n-1]$ and the previous mean squared error (MSE) $\boldsymbol{P}[n-1|n-1]$. Writing the forward state operator $\boldsymbol{F}$ as

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Gamma} \end{bmatrix},$$

the state update can be written as:

$$\hat{\boldsymbol{\theta}}[n|n-1] = \boldsymbol{F}\hat{\boldsymbol{\theta}}[n-1|n-1], \tag{38}$$

$$\boldsymbol{P}[n|n-1] = \boldsymbol{F}\boldsymbol{P}[n-1|n-1]\boldsymbol{F}^H + \boldsymbol{C}_w[n], \tag{39}$$

where

$$\boldsymbol{C}_w[n] = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_{\tilde{g}}[n] \end{bmatrix}$$

where $\boldsymbol{C}_{\tilde{g}}[n]$ as in Eq. (33).

A block diagram of the RETINA algorithm is shown in Figure 9.

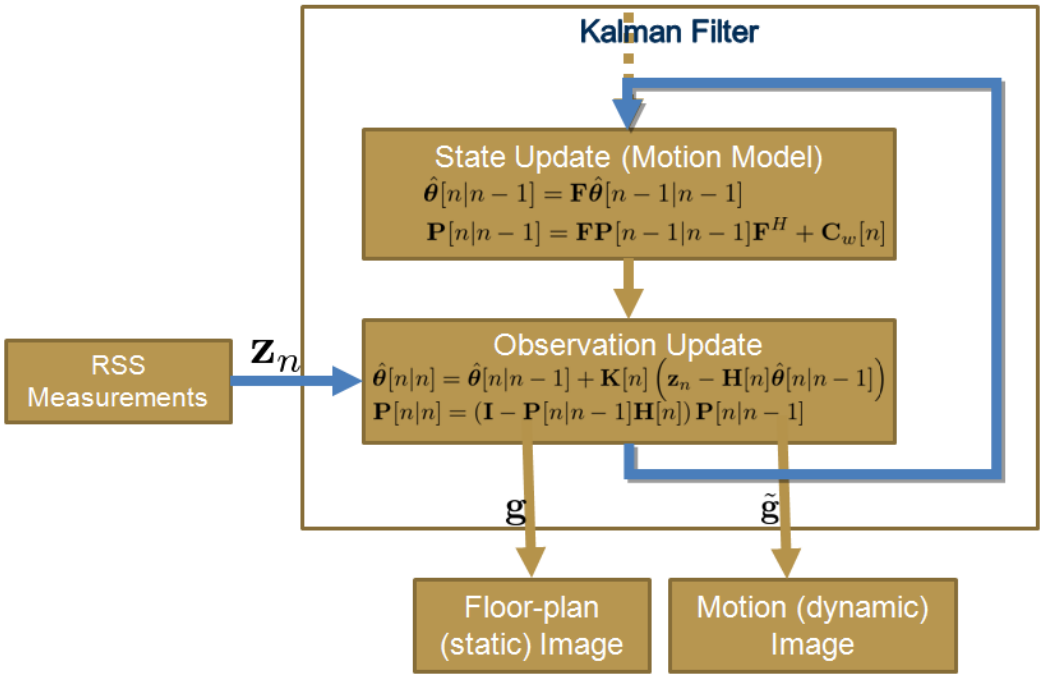


Figure 9: RETINA block diagram

3.4.2 Performance Comparison

We compared the performance of the RETINA algorithm to the modified variance-based algorithm described in Section 3.2 through simulation. The simulation consisted of 28 nodes around a simulated structure with 5 simulated objects moving inside the structure. Both algorithms were configured to estimate the spatial loss field for a 41x41 pixel grid centered on the simulated structure. The true spatial loss field for the structure is shown in Figure 10. The mobile objects were modeled as an additional spatial loss localized around the object's position. This additional spatial loss was in the shape of a small Gaussian with a variance of 1 pixel in size. Measurements were generated from the total spatial loss field using the NeSh model.

In the first simulation, the 28 measuring nodes were stationary and placed along the perimeter of the region. The reconstructed spatial loss fields after 3000 measurement/movement iterations for the variance-based algorithm and RETINA are shown in Figure 11. The filled circles denote the location of the measuring nodes, and the empty circles denote the location of the moving objects.

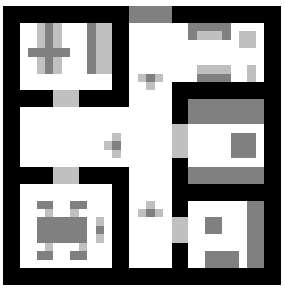


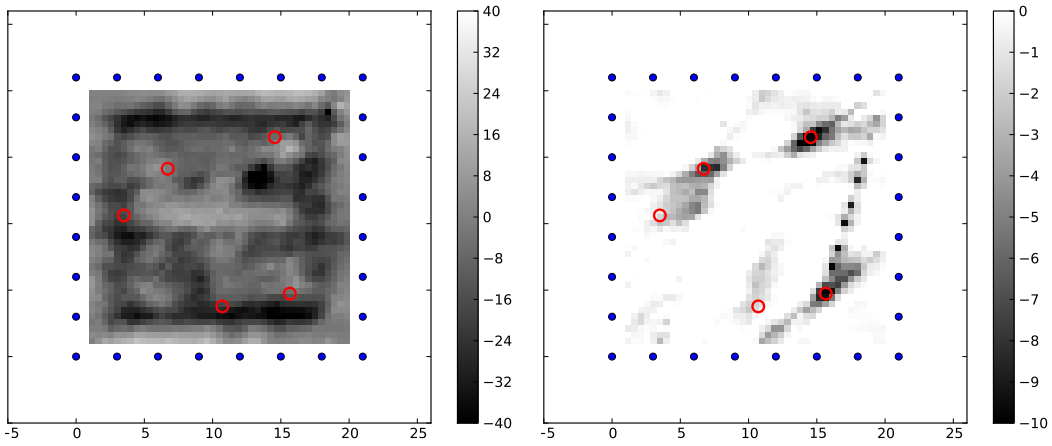
Figure 10: Simulated spatial loss field

Comparing Figure 11(a) and Figure 11(c), RETINA achieves comparable accuracy as the variance-based method for estimating the static structure. The reconstruction is not very clear because there are not enough unique measurement paths through the structure to detect its relatively complex shape.

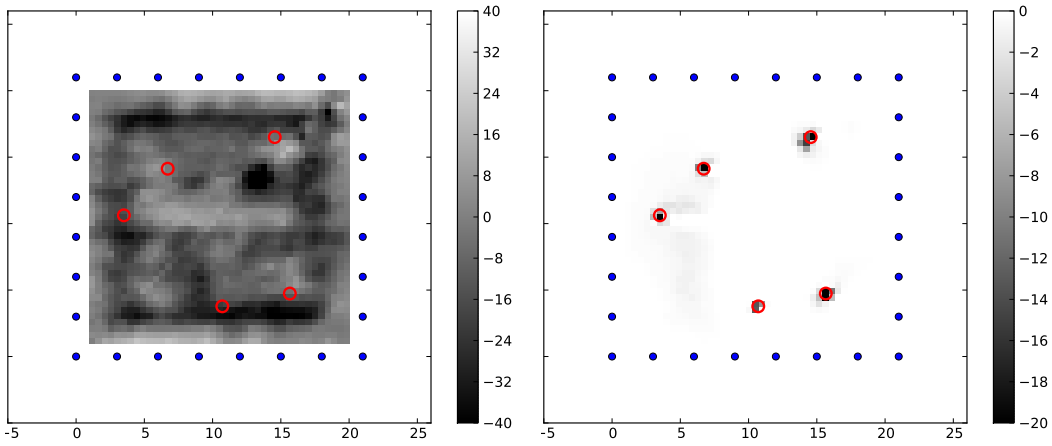
In contrast, there is a significant difference in the estimates of the moving objects. This can be seen by comparing Figure 11(b) and Figure 11(d). As expected from the analysis in Section 3.2, the variance-based method is only able to determine streaks where the moving object has been. If we instead consider the reconstruction provided by RETINA, it is apparent that RETINA has accurately and clearly detected the position of the moving objects.

In the second simulation, the 28 measuring nodes were placed randomly outside the imaging area and allowed to move slowly. The nodes moved according to a random velocity model such that they would pick a random velocity and travel at that velocity for a random duration. If they collided with either the boundaries of the network or the boundaries of the imaging area they would immediately pick a new velocity and duration. The velocity and duration were bounded such that nodes would move less than 0.1 units in each measurement interval. The current positions of all of the measurement nodes was assumed to be known by both RETINA and the variance-based algorithm. The reconstructed spatial loss fields after 400 measurement/movement iterations for the variance-based algorithm and RETINA are shown in Figure 12 along with the ending positions of the mobile measuring nodes.

The two reconstructions using the variance-based algorithm in Figure 12(a) and Figure 12(b)



(a) Variance-based static (b) Variance-based dynamic



(c) RETINA static (d) RETINA dynamic

Figure 11: Stationary reconstruction comparison

do not provide good estimates of either the static or dynamic aspects of the shadowing environment. While the analysis in Section 3.2 showed that this algorithm will perform poorly with mobile nodes, these results show that even slow movement can have catastrophic effects on the reconstruction.

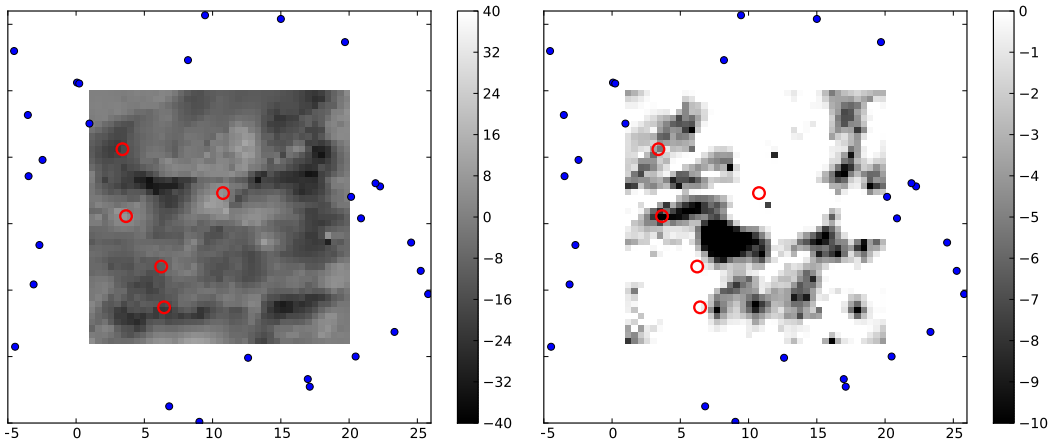
The reconstructions based on the RETINA algorithm, on the other hand, perform much better. The reconstruction of the static portion of the environment in Figure 12(c) shows that instead of losing accuracy when the measurement nodes are mobile, RETINA is able to take advantage of the additional information provided by the slightly different paths to significantly increase its resolving power. The additional resolving power allows RETINA to achieve a reconstruction comparable to the actual spatial loss field from Figure 10. The RETINA reconstruction dynamic portion of the spatial loss field also works quite well. Due to the motion of the moving objects, the additional paths can not be used to increase the resolving power. In spite of this, it was still able to track all 5 moving objects, and only suffered a few extra reconstruction artifacts.

3.5 Experimental Evaluation of Propagation Models

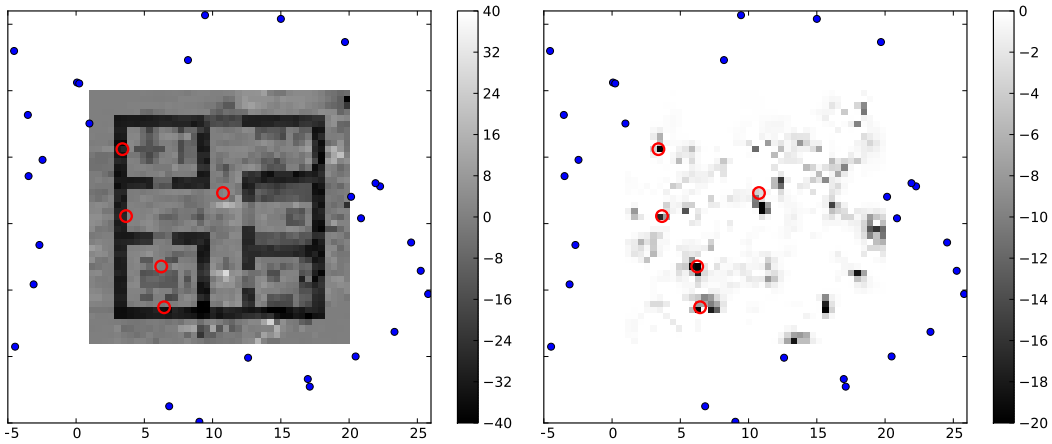
Next we turn to evaluating the accuracy of the RF propagation models available for RF tomography. While a couple of models have been presented, there has been little comparison or verification of these shadowing models. We present the existing models and propose a new model [60]. We describe the construction of our RF tomography testbed which we used to collect measurements from a realistic RF tomography scenario. We then evaluate how consistent each of the shadowing models are with the measurements from our field test.

3.5.1 Experiment Design

We designed our RF tomography testbed to emulate the conditions that would be expected in a practical application of RF tomography. The testbed consisted of two primary components, an artificial structure, and the channel measurement equipment. The key difference between this testbed and the testing done in previous works [2, 52] is that we are trying to determine the accuracy of the shadowing models instead of simply providing a proof of concept.



(a) Variance-based static (b) Variance-based dynamic



(c) RETINA static (d) RETINA dynamic

Figure 12: Mobile reconstruction comparison

3.5.1.1 Artificial Structure

The artificial structure was specifically designed to create a complex radio propagation environment. It was constructed out of plywood, sheetrock, concrete board and cinder block. In order to provide a more complicated shadowing environment the structure was designed with interior walls and an cinder block pillar in addition to the exterior walls. This provides a more complicated environment which should challenge the shadowing models we are testing. Each wall was composed of 3 modular 8' high by 4' wide panels supported by a 2x4 wood stud assembly. We constructed a total of 13 panels with the following breakdown:

- 3 3/8" plywood panels (exterior) with R11 insulation and sheetrock (interior)
- 3 1/2" OSB panels (exterior)
- 3 concrete board panels (exterior)
- 2 1/2" plywood panels (exterior)
- 1 1/2" plywood panel (interior)
- 1 sheetrock panel (interior)
- 1 column of cinder blocks

A diagram of the completed structure is shown in Figure 13.

3.5.1.2 Channel Measurement Equipment

Unlike previous works which only needed to demonstrate a proof of concept, we needed to make extremely accurate channel measurements so that we could separate errors in the tomographic reconstruction due to modeling error from those due to noise. Additionally, we needed to take significantly more channel measurements to detect the modeling error. In order to achieve both the accuracy and collection efficiency required, the channel measurements were collected using 10 Agilent E443x programmable signal generators as the transmitters and 10 Ettus USRP2 [61] software defined radios as the receivers. The flexibility of these devices allowed us to completely control the transmission and reception processes.

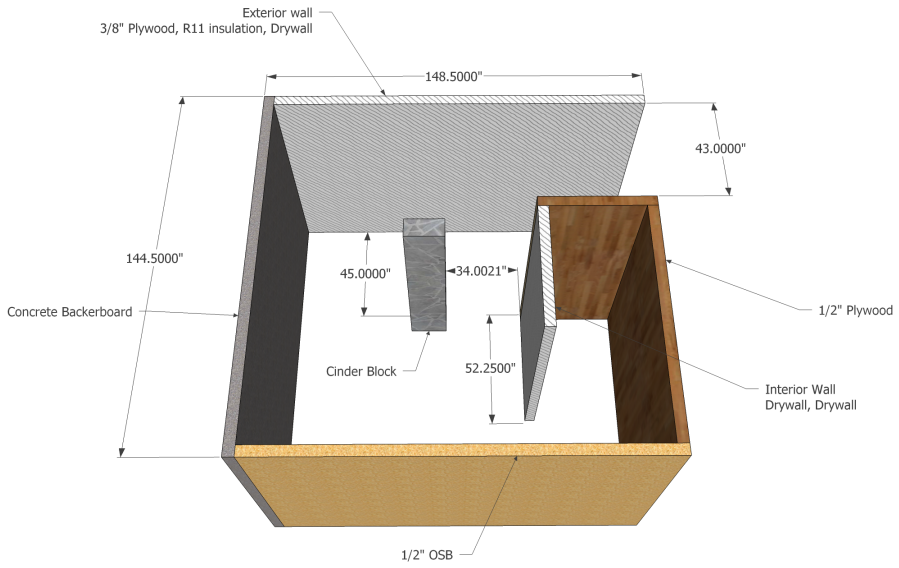


Figure 13: Diagram of shadowing structure

The signal generators serving as transmitters were programmed with channel sounding sequences designed for multi-user multiple input multiple output (MIMO) communication systems [62]. These sequences consist of a set of polyphase chirp waveforms which are all orthogonal and robust to differences in transmitter timing. In our testbed, the i^{th} transmitter was programmed with the sequence

$$s_i[n] = e^{j\pi 2^i \frac{n(n+1)}{N}},$$

for $i \in \{1, 2, \ldots, 10\}$ and $n \in \{0, 1, \ldots, N-1\}$. A plot of the time and frequency domain representations of this sequence for the 5^{th} transmitter is shown in Figure 14. Using orthogonal waveforms allowed us to use the 10 transmitters and 10 receivers to simultaneously estimate all 100 channels. Since these sequences are robust to differences in timing, we did not need to synchronize the transmission of these sequences. This allowed us to configure the the transmitters to transmit continuously throughout the entire field test without needing a complex triggering system to synchronize the transmitters and receivers. Further, since we used very long transmit sequences ($N = 2^{14}$ samples), we had an extremely high processing gain at the receiver, allowing us to obtain very high accuracy channel estimates. The channel measurement for the link from transmitter i to receiver j could then be extracted

from the received sequence $r_j[n]$ using a matched filter

$$\hat{h}_{i,j}[n] = \sum_{k=0}^{N} s_i[n-k] r_j[k],$$

where the indices are taken modulus N. The maximum of the magnitude of the channel estimate $\hat{h}_{i,j}[n]$ is then taken as the RSS estimate.

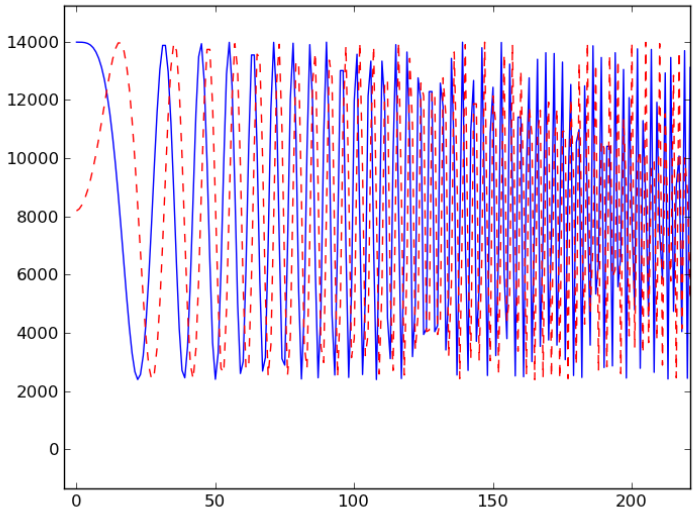
The software radios serving as the receivers were each connected over gigabit Ethernet to a laptop running Linux. The laptop was responsible for controlling the USRP2 radios and temporarily storing the raw samples of the received signal as it was being collected. The laptops in turn were controlled through a separate Ethernet network connecting them to the control computer. The control computer was connected to each laptop through Secure SHell (SSH) so that the control computer could trigger data collection and download the collected samples. This level of automation was necessary in order to collect a large number channel measurements in a limited period of time.

The outputs of the transmitters and inputs of the receivers were connected to 12dB omni-directional antennas through 20' long RF cables. These antennas were mounted on tripods which could be moved to measure the propagation characteristics between different parts of the network. The transmitters and receivers were also connected to rubidium atomic clocks as frequency references, to limit the error due to frequency offsets.

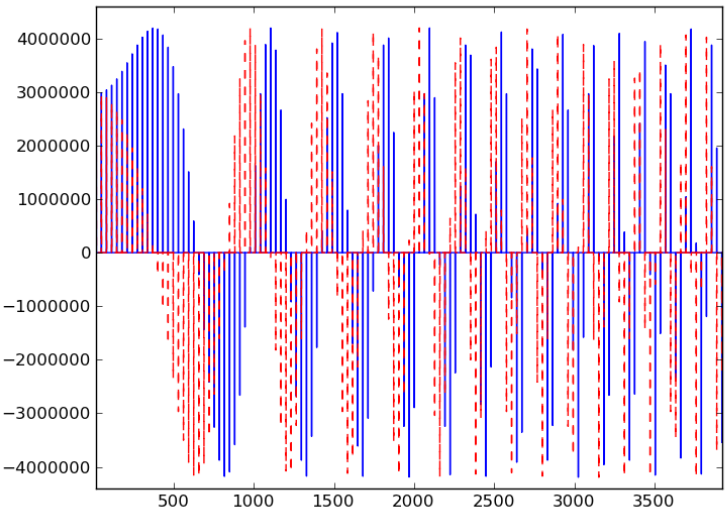
3.5.2 Data Collection

The equipment was divided onto two carts: Cart A, which hosted 6 transmitters and 6 receivers, and Cart B, which hosted 4 transmitters and 4 receivers. Each cart was connected to an independent EU3000 Honda generator to provide power. The control computer was co-located with Cart B and connected to the control network. The transmitters and receivers were configured so that channel measurements were taken at 2.425GHz with 5MHz bandwidth.

We measured the channel estimate for links between all combinations of positions on the perimeter of a 21x21 grid, a total of 2400 unique links. The grid spacing was 1', and the perimeter was around 4' from the structure. A diagram of the collection configuration is shown in Figure 15, where the solid 12'x12' block represents the artificial structure.



(a) Time domain



(b) Fourier transform

Figure 14: Waveform 5 example (real and imaginary parts)

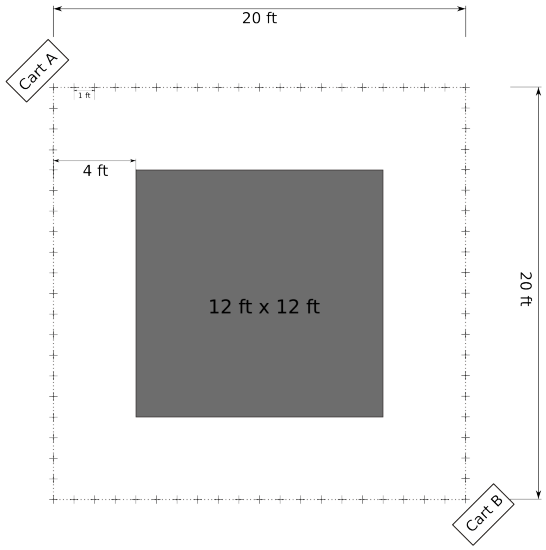


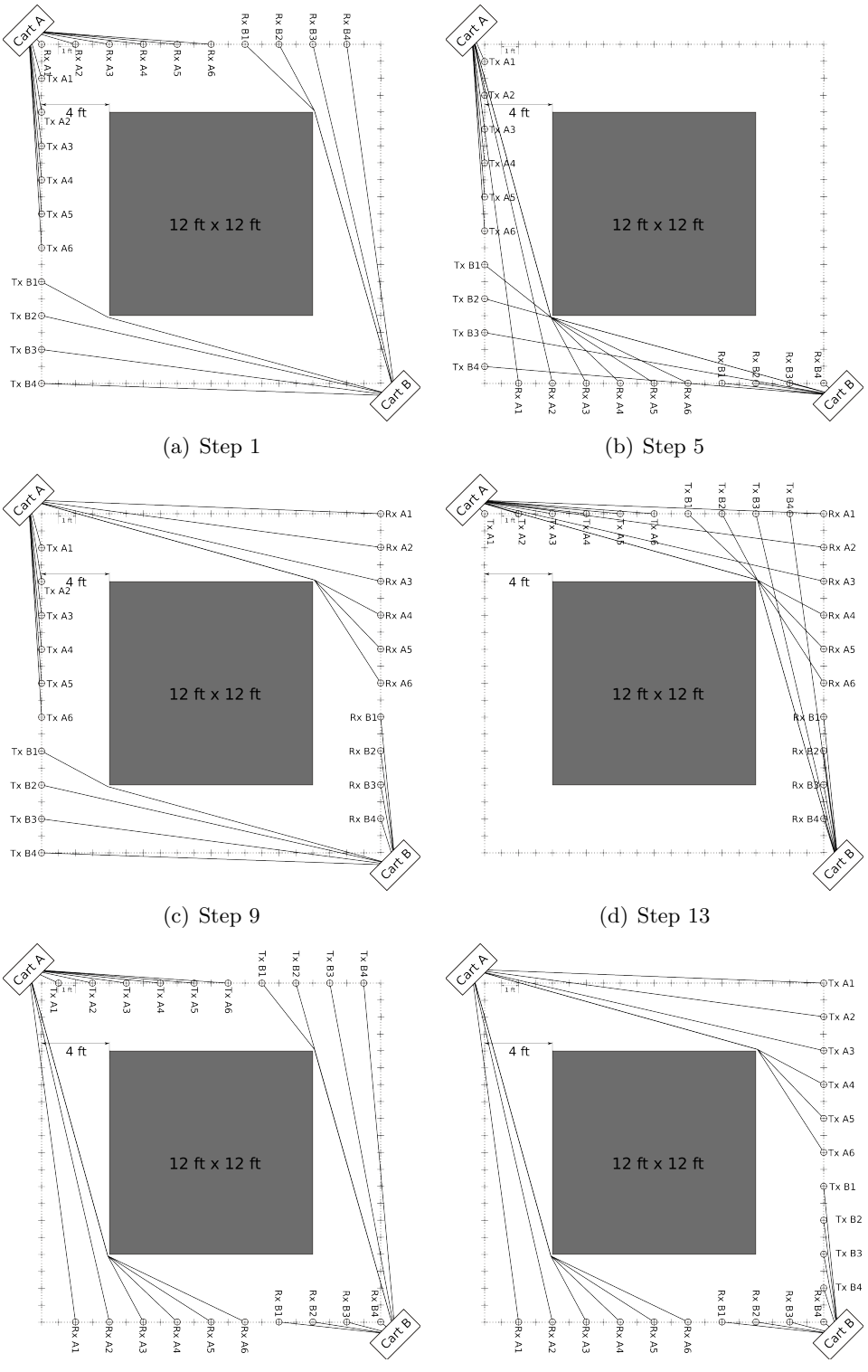
Figure 15: Collection layout

Since we were only using 10 receivers and 10 transmitters, the measurements were collected in a series of 24 steps. Between steps, either the receiver antennas or the transmitter antennas were moved to sound a different set of links. The most common movement was to simply shift the antennas 1' along the grid, however for several steps, the antennas had to be moved from one side of the grid to another. Diagrams of these larger movements are shown in Figure 16, where the circles show antenna positions, and the lines show how the cable was routed from the cart to the antenna. The data collected from all of these steps was composited into a single large data set which was then used for the reconstruction.

3.5.3 Tomographic Reconstruction

We performed a tomographic reconstruction using each of the models described in Section 3.3. As mentioned in Section 3.1, since the least-squares solution is not always fully determined, it is common to apply regularization techniques such as Tikhonov regularization to constrain the solution. We applied regularization based on the covariance from Eq. (22) with smoothing parameters σ_η^2 and κ.

We consider three different scenarios. In the "No Regularization" scenario, we employed a pure least-squared reconstruction to generate a 24x24 pixel image of the structure. For the "low smoothing" regularized case we set the parameters $\sigma_\eta = 0.2$ and $\kappa = 0.08$ feet



(a) Step 1

(b) Step 5

(c) Step 9

(d) Step 13

(e) Step 17

(f) Step 21

Figure 16: Collection steps

Table 2: Reconstruction SSR

	NeSh	Normalized Ellipse	Inverse Area Ellipse
No Regularization	8.59223×10^{12}	9.12867×10^{12}	7.24107×10^{12}
Regularized (low smoothing)	1.41001×10^{13}	9.98313×10^{12}	9.63178×10^{12}
Regularized (high smoothing)	2.21166×10^{13}	2.38467×10^{13}	1.93279×10^{13}

to generate a 60x60 pixel reconstructed image. With these parameters, the regularization had only a small smoothing effect on the reconstructed image. For the "high smoothing" regularized case we set the parameters $\sigma_\eta = 0.06$ and $\kappa = 0.08$ feet to generate a 30x30 pixel reconstructed image. These regularization parameters caused the resulting image to have significant smoothing effects from the regularization.

Before examining the reconstructed images, we will first review the sum of square residuals (SSR) from the reconstructions for each of the projection models. The vector of residuals for the reconstruction $\hat{\boldsymbol{g}}$ are given by $\boldsymbol{\rho} = \boldsymbol{B}\hat{\boldsymbol{g}} - \boldsymbol{\eta}$. Since these residuals will have lower magnitude when the model $\boldsymbol{B}$ is a better fit for the underlying process, we can use the SSR to show how closely the shadowing model matches the observations. Because our testbed was designed specifically to reduce the noise, the primary contributor to the SSR will be mismatch between the shadowing model and the radio propagation described by the measurements. Note that because regularization changes the optimization to put less emphasis on the least-square residuals, the SSR will increase as the weight of the regularization parameter increases.

The SSR for the reconstruction for these scenarios is shown in Table 2. From the table it is apparent that the Inverse Area Ellipse model has a significantly lower SSR than the other methods regardless of the scenario. Comparing the NeSh and Normalized Ellipse models, NeSh has a lower SSR for the "No Regularization" and "high smoothing" cases, but a higher SSR for the "low smoothing" case.

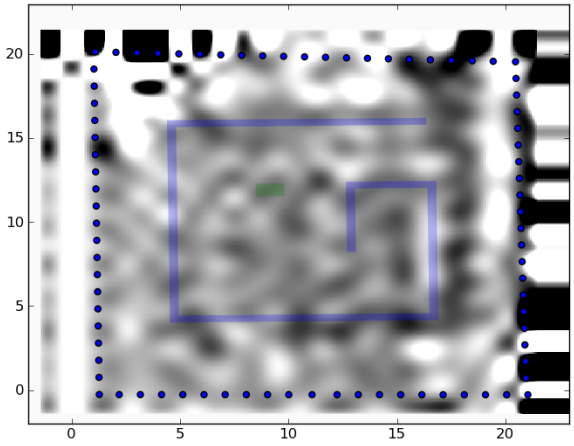
Of course the SSR can only show how well the model matches the underlying process. It is possible that the model that produces the best reconstruction for RF tomography may

not be the same as the model that matches the underlying process. We examine this aspect of the problem by comparing the reconstructed images and seeing how well these match against the shape of the structure.

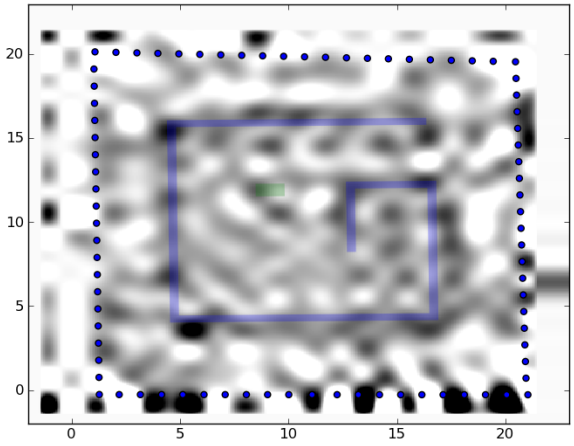
Figure 17 shows the reconstructed images for the "No Regularization" scenario. For clarity, we have overlaid a diagram of the actual position of the artificial structure in these images. From the figures, the reconstruction using the NeSh model clearly does not correspond to the test structure. In contrast, the Normalized Ellipse and Inverse Area Ellipse models have a reasonable correspondence to the structure that we were trying to image. Note that, while the NeSh model had a lower SSR than the Normalized Ellipse model for this scenario in Table 2, the actual reconstructed image was not useful for RF tomography.

The reconstructed images using regularization from the "low smoothing" scenario, shown in Figure 18 are able to provide more resolution. This does not significantly improve the reconstruction from the NeSh model, shown in Figure 18(a). However, in the Normalized Ellipse reconstruction from Figure 18(b), the rough outline of the building can be roughly distinguished, despite the noise-like artifacts obscuring the image. The Inverse Area Ellipse model's reconstruction (shown in Figure 18(c)), on the other hand, provides a relatively clear reconstruction of the structure with fewer artifacts. If we compare this result to the corresponding SSR from Table 2, the Inverse Area Ellipse model provides a much better tomographic reconstruction despite having a SSR only slightly better than the Normalized Ellipse model.

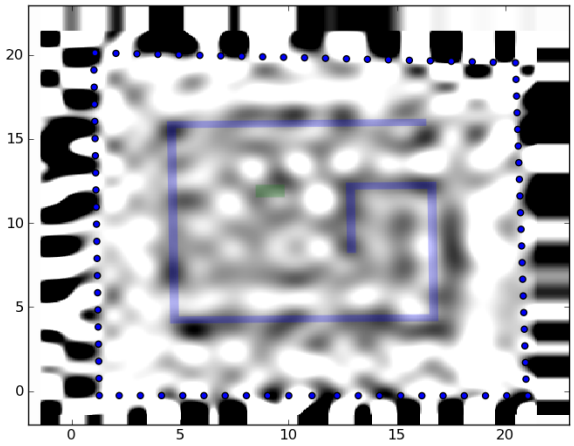
For the "high smoothing" scenario however, the Normalized Ellipse model appears to generate the best tomographic reconstruction despite of having the highest corresponding SSR in Table 2. This is shown in Figure 19. The NeSh model is still unable to recover the structure of the object in the shadowing environment. The Inverse Area Ellipse reconstruction recovers the basic shape, but the edges are not nearly as sharp as the Normalized Ellipse reconstruction. This makes sense, because the weights for the Inverse Area Elliptical tomographic projection model are more continuous than those of the Normalized Ellipse model. For this reason, the Inverse Area Elliptical reconstruction does not need the additional smoothing to obtain a useful reconstruction. Further the high SSR associated with



(a) NeSh

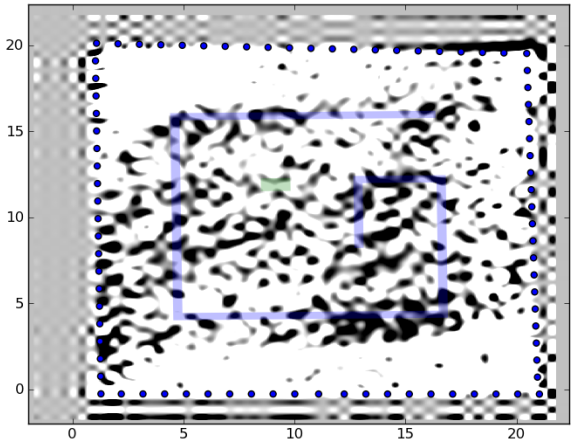


(b) Normalized Ellipse

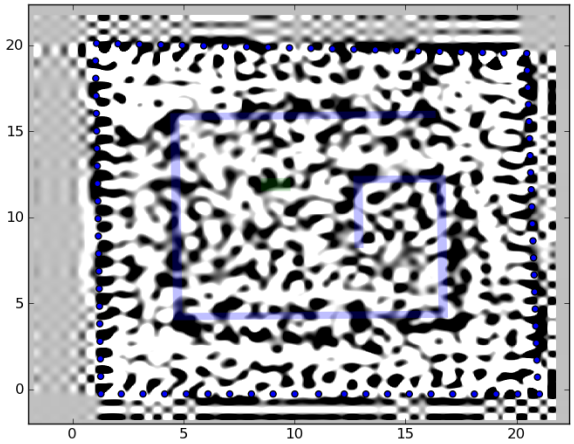


(c) Inverse Area Ellipse

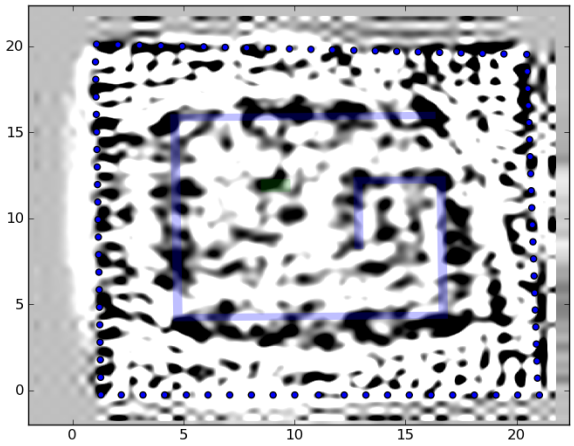
Figure 17: Reconstructions, no regularization



(a) NeSh



(b) Normalized Ellipse



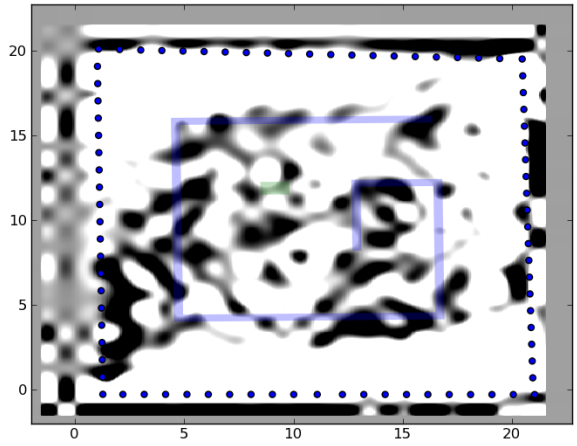
(c) Inverse Area Ellipse

Figure 18: Reconstructions, low smoothing

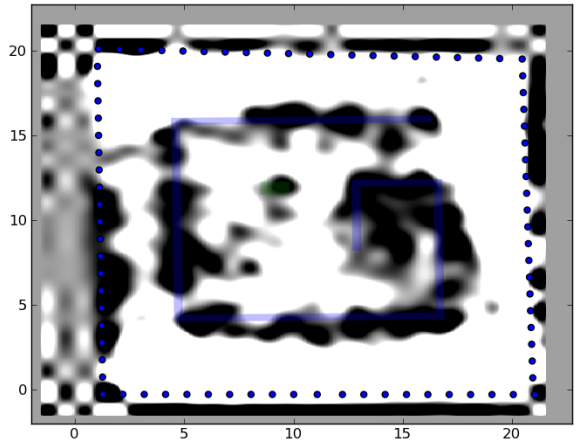
the Normalized Ellipse reconstruction suggests the regularization is being heavily relied upon to compensate for artifacts caused by the more binary projection used by the Normalized Ellipse model. While this allows the Normalized Ellipse model to create a useful reconstruction, the higher reliance on regularization suggests that the Inverse Area model may be a better model of the shadowing environment.

3.6 Concluding Remarks

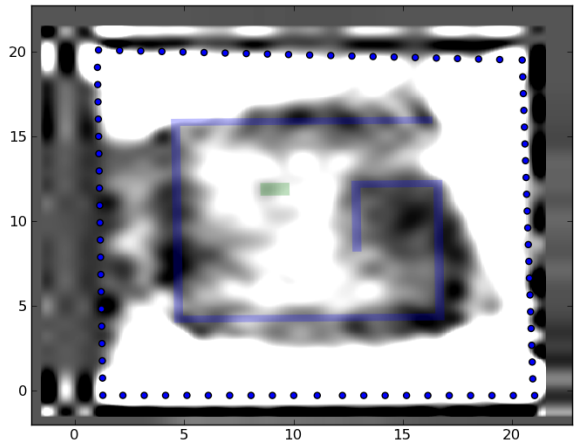
Ad-hoc networks of wireless devices have enormous potential. In this chapter we have described how such networks of wireless devices can opportunistically use the RSS measurements being collected to sense the physical environment the network is operating in. We presented our new RETINA algorithm and have shown that, in simulations, RETINA outperforms existing methods. We have described the design of our RF tomography testbed which we used in our field test. Our field test measurements were used to reconstruct the testbed environment, demonstrating the viability of this technique. We also compared several propagation models and have shown that our Inverse Area Elliptical model is a better model for the shadowing environment experienced in our field test.



(a) NeSh



(b) Normalized Ellipse



(c) Inverse Area Ellipse

Figure 19: Reconstructions, high smoothing

CHAPTER IV

WIRELESS LOCALIZATION

Location awareness can greatly enhance the capabilities of ad hoc and wireless sensor networks in a wide range of applications ranging from intrusion detection to environmental monitoring [7]. While, for some of these applications, nodes could be manually placed in surveyed locations, in many cases tedious placement of nodes is impossible, or the nodes may be mobile. For this reason, many applications require nodes in the network to be able to determine their own location. Additionally, nodes capable of self-localization can also be used to offer location-based services or to improve coordination between first-responders at disaster sites or infantry in tactical situations. Traditional localization techniques such as GPS may not be available due to power constraints, obstructions such as buildings, or interference such as that caused by hostile jamming.

4.1 Problem Statement

Localizing nodes in wireless networks is a challenging problem. Localization techniques generally classify nodes in the network as either reference nodes (also called anchor nodes), which know their location (using GPS or some other external location reference) and floating nodes which do not. Each type of node is able to take measurements such as angle, distance, delay or received signal strength (RSS). The localization problem is then to estimate the parameters of the network model that maximize the likelihood of obtaining the recorded measurements. The network model contains both a measurement model to determine the likelihood of a measurement being recorded given the network state and a movement model to describe how the position estimates and other network state change over time. We will consider two different network models for localization with RSS measurements.

We first consider the localization problem in a network of mobile floating nodes that are capable of taking accelerometer readings, where the RSS measurements are primarily affected by node distance. We propose to use an extended Kalman filter to estimate and

track the network state. We also describe a version of this algorithm which is distributed over the nodes in the network to improve the algorithm's scalability. Next we consider the localization problem in a network with both mobile reference and floating nodes where the RSS measurements are significantly affected by shadowing. We use a measurement model that includes a model of the shadowing in the network and propose to use distributed particle filters for localization. Finally, we investigate strategies to improve distributed estimation at a cost of increased communication overhead.

4.2 Existing Works

Localization techniques have been presented to use angle, distance or delay measurements [63–65], but these techniques require specialized sensors such as directional antennas, range-finders, or extremely accurate clocks that increase both the cost of the devices and their power requirements. Since systems using either connectivity or RSS measurements do not require specialized measurement hardware, several works have focused on using these measurements.

Methods using connectivity information, such as hop counts include LSVM [66], which uses the support vector machine learning method to determine node locations and Sequential Montecarlo Localization [67,68] which uses a particle filter based on connectivity information to track node positions. These methods generally assume a unit circle model of network connectivity, where all nodes within a certain distance of a given node are its neighbors. In practice, network connectivity has a significant random component, so these connectivity based approaches have limited resolution in all but the densest networks.

Further work has focused on using RSS measurements. Several techniques have been proposed using algorithms such as extended Kalman filters to [69], Semidefinite Programming [70], and Probability-based Maximum Likelihood Estimation [71]. The Cramér Rao bound (CRB) for the location estimation accuracy of these methods has also been calculated [72–74]. These techniques generally consider static networks where nodes are immobile. In networks with mobile nodes, these methods need to be modified and have difficulty tracking mobile nodes. Further, these techniques generally assume the RSS measurements

are a noisy function of distance. Since these methods do not consider the shadowing and fading effects of a realistic wireless channel which affect the RSS measurements, they have been shown to have relatively low accuracy [75]. Additionally, recent works [76] have indicated that the correlation from shadowing can be leveraged to further improve location accuracy.

Several works have proposed to use site surveys to improve accuracy when locating a single mobile node. These works [77–81] generally use site surveys to measure the channel from fixed base stations to various locations on site to generate a database of potential locations. Nodes are then located by interpolating between the measured locations. These techniques can provide good models of the network, however, they require the base stations to be located in the same position during the survey and localization phases, and can not readily be extended to multi-hop networks.

4.3 Network Models

We consider networks consisting of N_r reference nodes and N_m floating nodes distributed in a two dimensional region such that the density of floating nodes is D. This network consists of reference nodes with globally known locations and floating nodes with unknown locations. Mobile nodes move according to the movement model described in Section 4.3.1.

4.3.1 Movement Models

The nodes in the network may move according to one of the many movement models presented in the literature [82]. While we only specifically address two movement models here, the presented algorithms can be adapted to any desired mobility model. We consider both a simple AR position model which we will use to derive our localization algorithms, and a somewhat more realistic random velocity model which we use in simulations.

In the AR position model, node position is modeled as an autoregressive (AR) random process. The position is assumed to have a state equation such that $\boldsymbol{s}[t+1] = \boldsymbol{s}[t] + \boldsymbol{\psi}[t]$, where $\boldsymbol{s}[t]$ is the position at time t and $\boldsymbol{\psi}[t]$ is a zero mean Gaussian random noise vector with covariance $\sigma_{\psi}^2 \boldsymbol{I}$.

Since the random waypoint model has been shown to suffer from speed decay, where

the average speed of nodes approaches 0 as time progresses [83], we instead use a random velocity model. Our random velocity model is similar to the random waypoint model [82]. In our random velocity model, mobile nodes select a random direction (uniformly from $-\pi$ to π radians), a random speed (uniformly from 0 to 1), and two random intervals (both 0 to $T_{\max} = 0.5$s). The first time length describes the duration the node will accelerate from its current velocity to the new velocity. The second describes the duration the node will have constant velocity before selecting a new random velocity. Mobile nodes that hit the bounds of the network have an appropriate acceleration applied to prevent their escape and immediately choose a new random velocity.

4.3.2 Measurement Models

We consider three measurement models: one where distance is measured, one that uses RSS in the absence of shadowing, and another that uses RSS with shadowing according to the Network Shadowing (NeSh) model [50, 51]. We also describe an acceleration measurement model for nodes that are capable of measuring their absolute acceleration.

Using distance measurements

In the first model, the node is assumed to simply record a noisy measurement of the distance:

$$\hat{d} = d + \gamma,$$

where $\hat{d}$ is the measurement, d is the actual distance, and γ is Gaussian distributed noise. The distance for a given link is calculated as the Euclidean distance between the nodes involved: $d_{nm} = \sqrt{(s_n^x - s_m^x)^2 + (s_n^y - s_m^y)^2}$, $\boldsymbol{s}_n = (s_n^x, s_n^y)$ is the position of node n. Nodes are assumed to be connected if the measured distance $\hat{d}$ is less than some maximum range R.

RSS without shadowing

Most common devices do not have specialized hardware capable of directly determining distance. Instead, the distance is usually calculated from the received signal strength (RSS) or the height of the peak of the correlation from the correlator in the

receiver. The primary difference between the two is that the correlation peak may be more resistant to interference.

The RSS and peak correlation measure the received signal magnitude. We model this received signal magnitude as a pure function of distance disturbed by additive Gaussian noise with zero mean and variance σ_γ^2. So the RSS and peak correlation can be calculated as $\hat{z} = d^{-\alpha} + \gamma$, where α is the distance attenuation exponent (nominally between 2 and 4). Note that the distance measurement model described previously can be thought of as simply a special case of this model with $\alpha = -1$.

The RSS strength measurement can used for localization in one of two ways, either the location can be estimated from the RSS directly or the RSS can be converted to an equivalent distance measurement. Using the RSS measurement directly has the advantage that disturbing noise is simply the additive noise γ. Since γ is zero mean and Gaussian, the least-squares (LS) estimate of the position will be unbiased. On the other hand, localization using RSS measurements may be unstable do to the highly nonlinear relationship between the position and RSS measurements. Converting the RSS measurements to distances, on the other hand, causes the noise to be non-Gaussian, non-zero mean and correlated with distance. This noise creates a bias in the distance, causing far away nodes to seem further away, but nearby nodes to appear nearer still. As a result, the LS estimate of the position based on these distances will be biased, but the bias will depend on the network topology. Despite this bias, algorithms using these distance estimates are preferable since they are more stable. For this reason, we will only use the second method in this work.

RSS with shadowing

In the case of shadowing, we assume a single-path channel between two nodes with both shadowing and path loss. This model can be extended to multi-path channels if only the first arriving path is considered, and the other delayed paths treated as noise. The transmitted signal x is received as:

$$y = hx + u,$$

where h is the wireless channel and u is additive white Gaussian noise (AWGN). The channel h is

$$h = 10^{\frac{\eta}{20}} d^{-\alpha},$$

where d is the distance between transmitter and receiver, α is the path loss exponent (nominally between 2 and 4), and η is the shadowing component. We assume the shadowing component has the model described in Section 3.3.

We assume the RSS measurements are proportional to the channel estimate. In practical systems the proportionality constant can be determined and eliminated, so without loss of generality we model the RSS measurement $z_{i,j}$ from node i to node j as:

$$z_{i,j} = h_{i,j}(\boldsymbol{g}) + v_{i,j}, \tag{40}$$

where $v_{i,j}$ is AWGN with variance σ_v^2, and the channel $h_{i,j}(\boldsymbol{g})$ is

$$h_{i,j}(\boldsymbol{g}) = 10^{\frac{\boldsymbol{b}_{i,j}\boldsymbol{g}}{20\sqrt{d_{i,j}}}} d_{i,j}^{-\alpha}. \tag{41}$$

Acceleration Measurements

Many common portable wireless devices have begun incorporating acceleration sensors. The readings from these sensors can be combined with either readings from a rotation sensor or compass to measure the absolute acceleration. The methods for achieving this are beyond the scope of this work. Instead we assume that nodes directly measure the absolute acceleration with additive Gaussian noise η with variance σ_η^2: $\hat{a} = a + \eta$.

4.4 Location Tracking with Acceleration Measurements

In networks of devices capable of making acceleration measurements, such measurements could be used to improve localization performance. We propose a hybrid location tracking system which uses an extended Kalman filter to track the position and velocity of the mobile nodes [84]. In this algorithm, the acceleration measurement from the accelerometer will be added as a control input to the extended Kalman Filter, with the variance from this measurement being used as the state noise. We assume that, since accelerometer

measurements require significantly less power than distance measurements, nodes check the accelerometer some k times in the interval between synchronizations over the wireless interface, and that the interval between these checks is T. An alternative formulation would be to track the acceleration as part of the system state, but since the distance synchronization period should be large enough that the correlation in acceleration between these periods is quite small, we can avoid the complexity of additional state variables while still achieving similar performance.

4.4.1 Hybrid Location Tracking Algorithm

The extended Kalman filter consists of two main parts: the state update and innovation update. In the state update, the state update matrix $\boldsymbol{F}$ is applied to the previous state $\boldsymbol{\theta}[t|t]$ and the previous mean squared error (MSE) $\boldsymbol{E}[t|t]$. The control input $\boldsymbol{c}[t]$ is also added:

$$\boldsymbol{\theta}[t+1|t] = \boldsymbol{F}\boldsymbol{\theta}[t|t] + \boldsymbol{c}[t], \tag{42}$$

$$\boldsymbol{E}[t+1|t] = \boldsymbol{F}\boldsymbol{E}[t|t]\boldsymbol{F}^H + \boldsymbol{W}. \tag{43}$$

For our filter, the state $\boldsymbol{\theta}$ consists of the position (s^x, s^y) and velocity (v^x, v^y) of all the mobile nodes in the network:

$$\boldsymbol{\theta} = \begin{bmatrix} s_0^x & s_0^y & v_0^x & v_0^y & \cdots & s_{N_m-1}^x & s_{N_m-1}^y & v_{N_m-1}^x & v_{N_m-1}^y \end{bmatrix}^T.$$

The innovation update consists of incorporating the measurement with the current estimate. The measurement $\hat{\boldsymbol{z}}$ is assumed to be a function $h(\boldsymbol{\theta})$ of the state plus some noise $\boldsymbol{\gamma}[t]$ (with covariance $\boldsymbol{R}_\gamma[t]$):

$$\hat{\boldsymbol{z}}[t] = h(\boldsymbol{\theta}[t|t]) + \boldsymbol{\gamma}[t]$$

In the case of localization, the measurement vector contains the individual measurements for each of the L links in the network: $\hat{\boldsymbol{z}} = \begin{bmatrix} z_0 & z_1 & \cdots & z_{L-1} \end{bmatrix}$. Each of these individual measurements are calculated as:

$$z_l = \left((s_n^x - s_m^x)^2 + (s_n^y - s_m^y)^2\right)^{\frac{-\alpha}{2}},$$

where the l^{th} link is between node n and node m.

$$[\boldsymbol{H}[t]]_{ik} = \begin{cases} -\alpha(s_n^x - s_m^x) z_k^{\frac{\alpha+2}{\alpha}} & \text{for link } k \text{ between nodes } n \text{ and } m \text{ where } s_n^x \text{ is the } i^{\text{th}} \text{ component of } \boldsymbol{\theta} \\ -\alpha(s_n^y - s_m^y) z_k^{\frac{\alpha+2}{\alpha}} & \text{for link } k \text{ between nodes } n \text{ and } m \text{ where } s_n^y \text{ is the } i^{\text{th}} \text{ component of } \boldsymbol{\theta} \\ 0 & \text{else} \end{cases} \tag{48}$$

Since the function $h(\boldsymbol{\theta})$ is nonlinear in this case, we approximate it with its Jacobian evaluated at the estimated state, forming the Jacobian matrix $\boldsymbol{H}[t]$. The observation can then be incorporated into the state estimate using the Kalman gain $\boldsymbol{K}[t]$ as a weighting factor.

$$\boldsymbol{S}[t] = \boldsymbol{H}[t]\boldsymbol{E}[t|t-1]\boldsymbol{H}[t]^H + \boldsymbol{R}_\gamma[t] \tag{44}$$

$$\boldsymbol{K}[t] = \boldsymbol{E}[t|t-1]\boldsymbol{H}[t]^H \boldsymbol{S}[t]^\dagger \tag{45}$$

$$\boldsymbol{\theta}[t|t] = \boldsymbol{\theta}[t|t-1] + \boldsymbol{K}[t]\left(\hat{\boldsymbol{z}}_m - h(\boldsymbol{\theta}[t|t-1])\right) \tag{46}$$

$$\boldsymbol{E}[t|t] = \left(\boldsymbol{I} - \boldsymbol{K}[t]\boldsymbol{H}[t]\right)\boldsymbol{E}[t|t-1] \tag{47}$$

For the localization problem, this Jacobian is shown in Eq. (48).

In order to determine the state update matrix $\boldsymbol{F}$, state covariance matrix $\boldsymbol{W}$, and the control input $\boldsymbol{c}$, we look briefly into how state changes due to a measured acceleration will affect the state transition from $\boldsymbol{\theta}[t|t]$ to $\boldsymbol{\theta}[t+1|t]$.

While the acceleration has a linear effect on velocity, its affect on position is nonlinear function of time. This means that we cannot directly integrate the measurements of acceleration as the control input. For simplicity, we consider the 1-dimensional case at a single node. Since each of the dimensions and each of the nodes progress independently, this derivation can be easily generalized to a full 2-dimensional network. Nodes use the kinematic equations to update their position each time they checks the accelerometer:

$$x[n] = x[n-1] + v[n-1]T + 0.5a[n]T^2$$
$$v[n] = v[n-1] + a[n]T,$$

where $x[n-1]$ and $v[n-1]$ are the current position and velocity estimates respectively, and $a[n]$ is the n^{th} acceleration measurement.

In matrix form, we can write:

$$\boldsymbol{x}[n] = \boldsymbol{F}_a \boldsymbol{x}[n-1] + \begin{bmatrix} \frac{a[n]T^2}{2} \\ a[n]T \end{bmatrix},$$

with $\boldsymbol{F}_a = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$.

Since $\boldsymbol{x}[n]$ is updated k times between each synchronization interval, we express $\boldsymbol{x}[k]$ in terms of $\boldsymbol{x}[0]$ as:

$$\boldsymbol{x}[k] = \boldsymbol{F}_a^k \boldsymbol{x}[0] + \sum_{m=0}^{k-1} \boldsymbol{F}_a^m \begin{bmatrix} \frac{a[k-m]T^2}{2} \\ a[k-m]T \end{bmatrix}. \tag{49}$$

If we let $\boldsymbol{x}[0]$ be a node's state in the Kalman filter state vector $\boldsymbol{\theta}[t|t]$ and $\boldsymbol{x}[k]$ be its state in $\boldsymbol{\theta}[t+1|t]$ and compare Eq. (49) to the state update equation used by the Kalman filter (Eq. (42)), we find that $\boldsymbol{F} = \boldsymbol{F}_a^k = \begin{bmatrix} 1 & kT \\ 0 & 1 \end{bmatrix}$ and $\boldsymbol{w}[n] = \sum_{m=0}^{k-1} \boldsymbol{F}_a^m \begin{bmatrix} \frac{a[n-m]T^2}{2} \\ a[n-m]T \end{bmatrix}$. The mean of the state noise $\boldsymbol{w}$ is then:

$$\bar{\boldsymbol{w}} = \begin{bmatrix} T \sum_{n=0}^{k-1} \frac{v[n]+v[n-1]}{2} \\ T \sum_{n=0}^{k-1} a[n] \end{bmatrix} \tag{50}$$

$$= \begin{bmatrix} T^2 \sum_{n=1}^{k-1} \left(\frac{a[n]}{2} + (k-n)a[n-1] \right) \\ T \sum_{n=0}^{k-1} a[n] \end{bmatrix}, \tag{51}$$

and its covariance becomes:

$$\boldsymbol{W} = \sigma_\eta^2 T^2 \begin{bmatrix} T^2(\frac{k^3}{3} - \frac{k}{12}) & T\frac{k^2}{2} \\ T\frac{k^2}{2} & k \end{bmatrix}.$$

We use the mean of the state noise from Eq. (51) as the control input $\boldsymbol{c}[t]$. Instead of performing the summation, we maintain an accumulator $\boldsymbol{c}_a$ which keeps a running total of the offset in position and velocity due to the acceleration measurements. This vector is updated according to:

$$\boldsymbol{c}_a[n+1] = \boldsymbol{c}_a[n] + \begin{bmatrix} \boldsymbol{a}[n]T^2(k-n+0.5) \\ \boldsymbol{a}[n]T \end{bmatrix},$$

where $\boldsymbol{a}[n]$ is the measured acceleration at the n^{th} time the accelerometer was checked during the current interval and $\boldsymbol{c}_a[0] = \mathbf{0}$. This accumulator is then used as the control input (i.e. $\boldsymbol{c} = \boldsymbol{c}_a[k]$) each time the node synchronizes over the wireless interface.

Note however that this technique will tend to underestimate the error due to the observations of the acceleration. The true acceleration is a continuous process, and only the instantaneous acceleration is measured. This measurement is used as the average acceleration over the entire interval between checks. Since the acceleration actually changes during this interval, there will be an additional amount of error. If the checking interval is made sufficiently small, the acceleration will be roughly constant, and this error will become negligible.

4.4.2 Distributed Implementation

The calculation of $\boldsymbol{K}$ in Eq. (45) for a large network of nodes can be very computationally intensive. To reduce the computational cost, we split the problem and solve each node separately using the estimated positions of the other nodes as reference nodes. The Kalman filter update equations for node m, then become:

$$\boldsymbol{\theta}_m[t+1|t] = \boldsymbol{F}_m\boldsymbol{\theta}_m[t|t] + \boldsymbol{c}_m[t], \tag{52}$$

$$\boldsymbol{E}_m[t+1|t] = \boldsymbol{F}_m\boldsymbol{E}_m[t|t]\boldsymbol{F}_m^H + \boldsymbol{W}_m. \tag{53}$$

$$\boldsymbol{S}_m[t] = \boldsymbol{H}_m[t]\boldsymbol{E}_m[t|t-1]\boldsymbol{H}_m[t]^H + \boldsymbol{R}_\gamma[t] \tag{54}$$

$$\boldsymbol{K}_m[t] = \boldsymbol{E}_m[t|t-1]\boldsymbol{H}_m[t]^H\boldsymbol{S}_m[t]^\dagger \tag{55}$$

$$\boldsymbol{\theta}_m[t|t] = \boldsymbol{\theta}_m[t|t-1] + \boldsymbol{K}_m[t]\left(\hat{\boldsymbol{z}}_m - h(\boldsymbol{\theta}_m[t|t-1])\right) \tag{56}$$

$$\boldsymbol{E}[t|t] = \left(\boldsymbol{I} - \boldsymbol{K}_m[t]\boldsymbol{H}[t]\right)\boldsymbol{E}[t|t-1] \tag{57}$$

This can be viewed as a distributed filter implementation of the hybrid location tracking algorithm.

One problem with treating all of the neighboring nodes as if they are reference nodes is that it does not consider the error in the node's position estimate, causing it to underestimate the resulting error in the the location estimate. This location error will cause the

Kalman filter to converge to a suboptimal location estimate. We will describe how this algorithm can be modified to avoid this problem in Section 4.6.

4.4.3 Modified Multilateration

For comparison purposes we define a trivial modification of the Multilateration algorithm [69, 85] allowing it to use a full extended Kalman filter to track the position. The original method assumed a static network, and only used the innovation update portion of the Kalman filter.

We extend this into a full extended Kalman filter by incorporating the AR position movement model described in Section 4.3.1. This means that the state update equation will be very similar to Eq. (52) in our distributed hybrid location tracking algorithm:

$$\boldsymbol{\theta}_m[t+1|t] = \boldsymbol{F}\boldsymbol{\theta}_m[t|t] + \boldsymbol{\psi}_m[t],$$

where the state $\boldsymbol{\theta}_m$ only contains node positions, $\boldsymbol{F}$ is the identity matrix, and $\boldsymbol{\psi}_m[t]$ is some the AR position state noise with covariance $\sigma_\psi^2 \boldsymbol{I}$.

We will compare our hybrid location tracking algorithm to this method in Section 4.4.4 and Section 4.4.5.

4.4.4 Theoretical Bounds on Performance

In [86] the Posterior Cramér Rao Bound (PCRB) for nonlinear filtering was proposed. In this section we apply this bound to the proposed tracking algorithms. The Posterior Cramér Rao Bound (PCRB) [86] provides a lower bound for the mean-squared error (MSE) due to estimation or a non-linear dynamic system. This error lower bounded by the diagonal elements of the inverse of the Fisher information matrix $\boldsymbol{J}$. Since we assume the state and observation noise are both additive Gaussian and the state update is linear, we can express the Fisher information matrix for the system state at time $t+1$ based on the first t observations using the recursion:

$$\boldsymbol{J}[t+1] = \boldsymbol{D}^{22}[t] - \boldsymbol{D}^{21}\left(\boldsymbol{J}[t] + \boldsymbol{D}^{11}\right)^{-1}\boldsymbol{D}^{12},$$

where

$$\boldsymbol{D}^{11} = \boldsymbol{F}^H \boldsymbol{W}^{-1} \boldsymbol{F}$$

$$\boldsymbol{D}^{12} = -\boldsymbol{F}^H \boldsymbol{W}^{-1}$$

$$\boldsymbol{D}^{21} = \left(\boldsymbol{D}^{12}\right)^H$$

$$\boldsymbol{D}^{22}[t] = \boldsymbol{W}^{-1} + \boldsymbol{H}[t]^H \boldsymbol{\Gamma}^{-1} \boldsymbol{H}[t].$$

The PCRB for the covariance of the estimate of the system state at time $t+1$ is then given as $\boldsymbol{J}[t+1]^{-1}$. In the case where the matrix $\boldsymbol{H}[t]$ is relatively constant and the system has reached a steady-state, the steady state estimation error can be calculated as the solution for $\boldsymbol{J}_\infty$ in the Riccati equation:

$$\boldsymbol{J}_\infty = \boldsymbol{D}^{22} - \boldsymbol{D}^{21} \left(\boldsymbol{J}_\infty + \boldsymbol{D}^{11}\right)^{-1} \boldsymbol{D}^{12}.$$

This solution can be found numerically in MatLab with the 'dare' function. The steady-state PCRB is: PCRB $= \boldsymbol{J}_\infty^{-1}$.

We apply this PCRB to both the hybrid location tracking algorithm we described previously and the modified multilateration algorithm described in Section 4.4.3. The network (shown in Figure 20) was chosen to be similar to the hexagonal networks previously used [73] to evaluate the Cramér Rao Bound for localization in static networks. In order to avoid edge effects, we only consider the PCRB of the position of a single node located near the center of the network (marked in the figure with an 'x'). We examine the PCRB using both the distance ($\alpha = -1$) and the RSS ($\alpha = 2$) measurement models. Figure 21(a) shows the PCRB as a function of the observation noise power σ_w^2, and Figure 21(b) shows the PCRB as a function of the accelerometer noise power σ_η^2. From the plots it is apparent that the hybrid location tracking algorithm outperforms multilateration. When the observation noise power is high relative to the accelerometer noise power, the hybrid location tracking algorithm has a bound relatively independent of the observation noise. When the observation noise becomes small enough relative to the accelerometer noise power, the hybrid location tracking algorithm approaches the bound achieved by multilateration until they

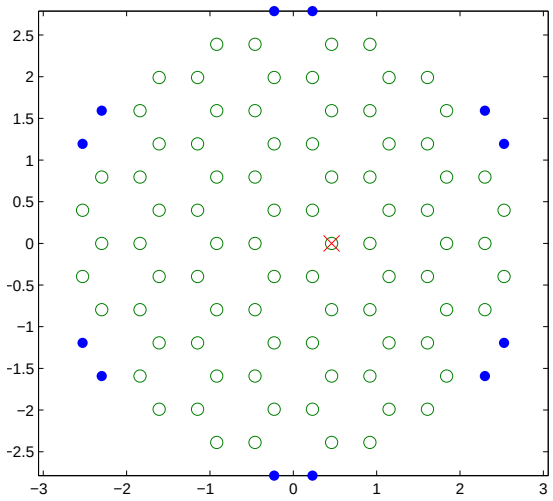


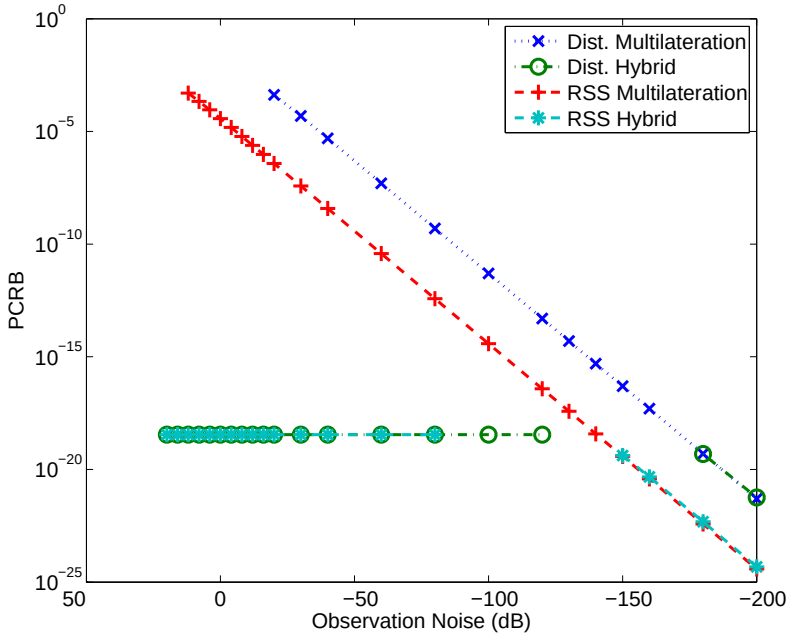
Figure 20: Network used to estimate PCRB. Reference nodes are solid circles. Mobile nodes are open circles. The open circle containing an 'x' represents the location where the PCRB shown in the following plots was taken.

are roughly collinear. This means that the hybrid location tracking algorithm is able to take advantage of the low noise power from either measurement source.

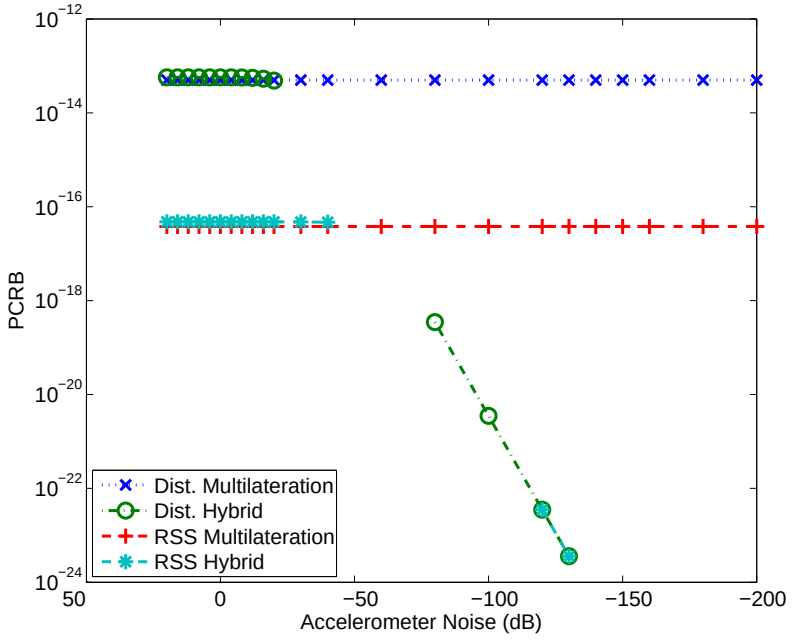
The two operating regions are also apparent in Figure 22. This figure contains two plots of the PCRB versus network density. In Figure 22(a), the observation noise is high relative to the accelerometer noise, so the hybrid location tracking algorithm's bound primarily determined by the accelerometer noise, and is relatively independent from network density. This is contrasted with Figure 22(b), where the hybrid location tracking algorithm and multilateration exhibit similar bounds. This once again shows that the hybrid location tracking algorithm is able to take advantage of the relatively higher accuracy available from the acceleration measurements to compensate for the higher accuracy RSS measurements.

4.4.5 Performance Comparison

We simulate our hybrid location tracking algorithm and the modified multilateration location tracking algorithm on a network with $N_m = 60$ mobile nodes and $N_r = 4$ reference nodes to compare the accuracy of the localization and tracking estimates. The mobile nodes are randomly placed in a square region. This region is sized such that the nodes have a

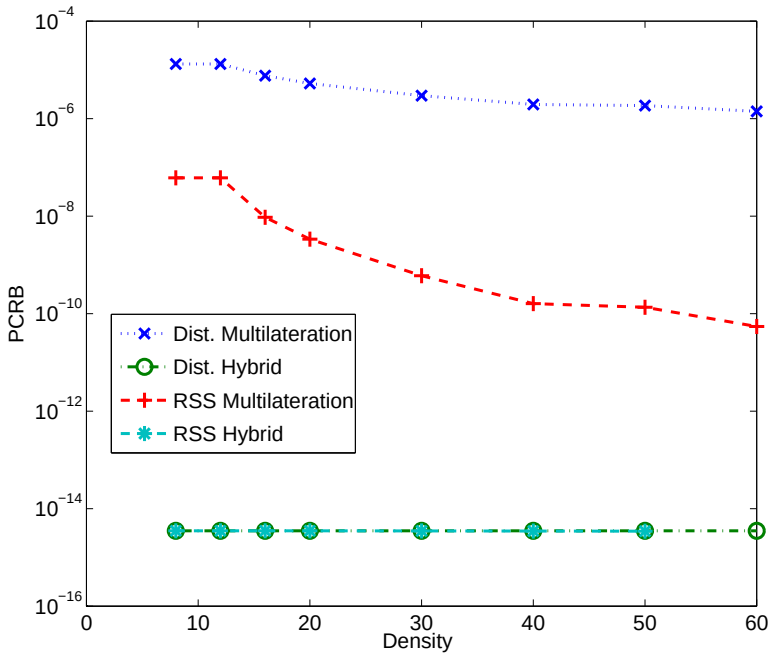


(a) PCRB vs. Observation Noise (σ_w^2 dB)

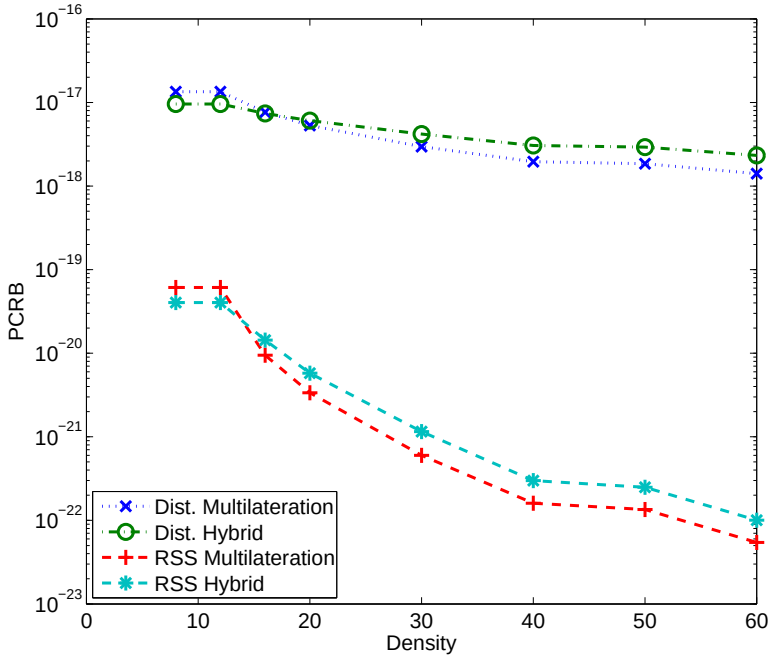


(b) PCRB vs. Accelerometer noise (σ_η^2 dB)

Figure 21: Steady-state PCRB



(a) High Observation Noise (-40dB)



(b) Low Observation Noise (-160dB)

Figure 22: Steady-state PCRB vs. density (nodes/ sq. unit)

density between 1 and 30 nodes per square unit. Each node is assumed to have a communication range of 1 unit, and be capable of making measurements to nodes within this distance. Reference nodes are placed at the 4 corners of the square.

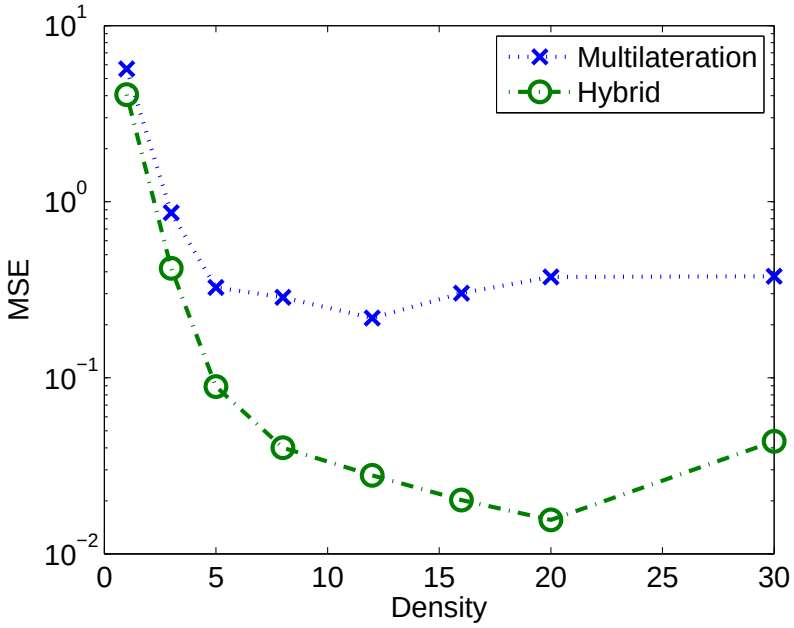
Mobile nodes move according to the random velocity model described in Section 4.3.1 Nodes check their acceleration every 0.0001 seconds and synchronize every 0.01 seconds. The simulation is run for 500 synchronization periods to allow it to converge. The results of several simulation runs are combined to estimate the performance at each of several noise levels and network densities.

Figure 23(a) shows the mean-squared error location tracking performance for 0dB noise power as density increases. Our hybrid location tracking algorithm is able to achieve relatively high performance despite the relatively large noise power. Additionally Figure 23(b) shows that our tracking algorithm maintains the lower MSE even when the noise power decreases.

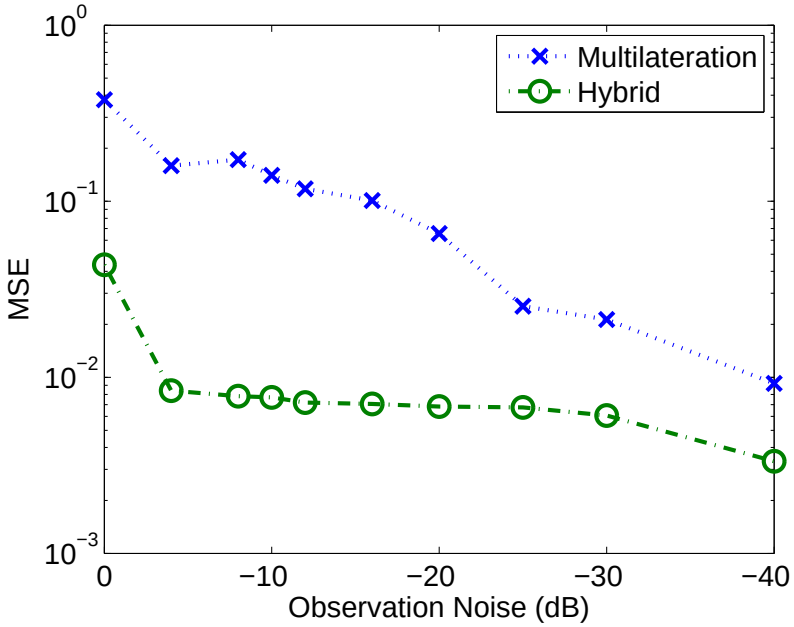
4.5 Shadowing Assisted Localization (SAL)

For networks where the effects of shadowing are significant, we propose the Shadowing Assisted Localization (SAL) algorithm [87]. This technique consists of an initial site survey phase and a localization phase. In the site survey phase, we construct a model of the shadowing environment. In the localization phase, mobile nodes use RSS measurements between themselves and either other floating nodes or anchor nodes to estimate the location of floating nodes in a multi-hop network. The shadowing model is used to along with a particle filter to estimate and track the positions of floating nodes. This technique differs from existing techniques in that it combines all of the following features:

1. Reference nodes can be mobile. Further, nodes can change between being reference nodes and floating nodes dynamically with no communication overhead.
2. Localization occurs over multiple hops, i.e. even floating nodes that do not have any reference nodes as neighbors can be localized.
3. SAL exploits knowledge of shadowing environment to improve localization accuracy.



(a) Position MSE vs Network Density (at 0dB observation noise)



(b) Position MSE vs Observation Noise (with density of 3)

Figure 23: Postion MSE

4.5.1 Site Survey

In the site survey phase, a model of the shadowing environment in the network area is constructed. Any linear shadowing model such as those described in Section 3.3 can be used. The model parameters can be estimated based on theoretical analysis such as ray-tracing the floorplan, or based on empirical measurements from a site survey.

The site survey consists of a series of RSS measurements made from nodes with known positions. These measurements are used to estimate a model for the shadowing in the network. Estimation of the shadowing model can be shown to be the RF tomography problem. Specific methods for forming the shadowing model from the site survey measurements has been discussed in Chapter 3.

4.5.2 Localization with Shadowing

Once the shadowing model has been generated from the site survey, it can be used to track the position of the nodes. We employ a particle filter [6] to track the floating nodes. Each floating node uses an independent particle filter to estimate its own location, and then all the nodes in the network broadcast their most recent position estimate to all of their neighbors. Since the only difference in behavior between the floating nodes and reference nodes is that the floating nodes use a particle filter to estimate their position and reference nodes use an external reference for the location estimate, all nodes need to do to change from reference nodes to floating nodes and vice versa is to change how they obtain their position estimates.

More explicitly, each floating node n independently forms an estimate of its own position $\boldsymbol{s}_n$. Each floating node has N_P particles which represent samples from its position distribution. Each particle has a position $\boldsymbol{s}_n^{(k)}$ and a weight $w_n^{(k)}$. The weight is proportional to the likelihood of that particle's position being correct. Floating nodes calculate an estimate of their locations $\boldsymbol{s}_n$ as the average of the locations of all of their particles, weighted by the particle weights:

$$\boldsymbol{s}_n = \sum_{k=0}^{N_P-1} \boldsymbol{s}_n^{(k)} w_n^{(k)} \tag{58}$$

At each measurement interval t, each floating node n receive beacons from each neighbor

m containing their neighbor's current position estimate $\boldsymbol{s}_m$. The node also measures the RSS of the beacon as $\hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m}$. The floating node first updates its particles according to the movement model. In the case of the AR position movement model described in Section 4.3.1, the particles are each disturbed by a zero mean Gaussian random noise vector with covariance $\sigma_\psi^2 \boldsymbol{I}$, and the weights are unchanged:

$$\boldsymbol{s}_n^{(k)}[t] = \boldsymbol{s}_n^{(k)}[t-1] + \boldsymbol{\psi}_n^{(k)}[t].$$

The node then performs the measurement update.

For the measurement update, the node first calculates the expected measurement $\tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}$ for each particle k. This expected measurement represents what the measurement would be if the node were actually located at the position of that particle. This expected measurement is calculated as in (40) using the particle's position $\boldsymbol{s}_n^{(k)}[t]$ for the position of node n and the most recent position estimate $\boldsymbol{s}_m$ for the neighboring node m:

$$\tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} = d_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}^{-\alpha} 10^{\frac{1}{20}\left(d_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}\right)^{-\frac{1}{2}}\left(\boldsymbol{b}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}\right)\boldsymbol{g}}. \tag{59}$$

The particle weights can then be updated as

$$w_n^{(k)}[t+1] = w_n^{(k)}[t] \prod_m q[n,k,m], \tag{60}$$

where $q[n,k,m]$ is the weight adjustment for particle k from the measurement from node m. This weight adjustment should be proportional to

$$\mathrm{P}\left\{z_{\boldsymbol{s}_n,\boldsymbol{s}_m} = \tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} | \hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m}\right\}.$$

Since we are assuming that both the measurement noise γ and the uncertainty in the expected measurement are Gaussian with the same mean, the weight adjustment $q[n,k,m]$ in Eq. (73) should be the Gaussian probability density function with variance σ_γ^2:

$$q[n,k,m] = \frac{1}{\sqrt{2\pi\sigma_\gamma^2}} e^{-\frac{\left(\hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m} - \tilde{z}_{\boldsymbol{s}_n^{(k)},m}\right)^2}{2\sigma_\gamma^2}}. \tag{61}$$

After all of the measurements have been processed, it is necessary to renormalize the particle weights so that

$$\sum_{k=0}^{N_P-1} w_n^{(k)}[t+1] = 1.$$

Since particle filters require resampling periodically to prevent the weights of the majority of particles from degenerating to 0, each node will periodically resample the particles. During resampling the particles are converted to an approximate probability distribution and random values are then sampled from that distribution to serve as the new particles. We model the probability distribution as a sum of N_P Gaussians centered on each particle such that the distribution has the probability density function:

$$p_n(x) = \sum_{k=0}^{N_P-1} \frac{w_n^{(k)}}{\sqrt{2\pi\sigma_s^2}} e^{-\frac{1}{2\sigma_s^2}\left(x-\boldsymbol{s}_n^{(k)}\right)^2},$$

where σ_s^2 is a configurable spread parameter. After resampling, all the particles have weight $\frac{1}{N_P}$ and are random samples from this distribution (i.e. $\boldsymbol{s}_n^{(k)} \sim p_n(x)$). As in [6], resampling is triggered when the effective number of particles,

$$N_{\text{eff}} = \left(\sum_k {w_n^{(k)}}^2\right)^{-1},$$

falls below a set threshold, N_{thr}.

Initially, the particles are assumed to be uniformly distributed over the field with equal weight. This algorithm continues iteratively as often as necessary to achieve the desired accuracy and for as long as location estimates are required. A summary of this algorithm is shown in Algorithm 1.

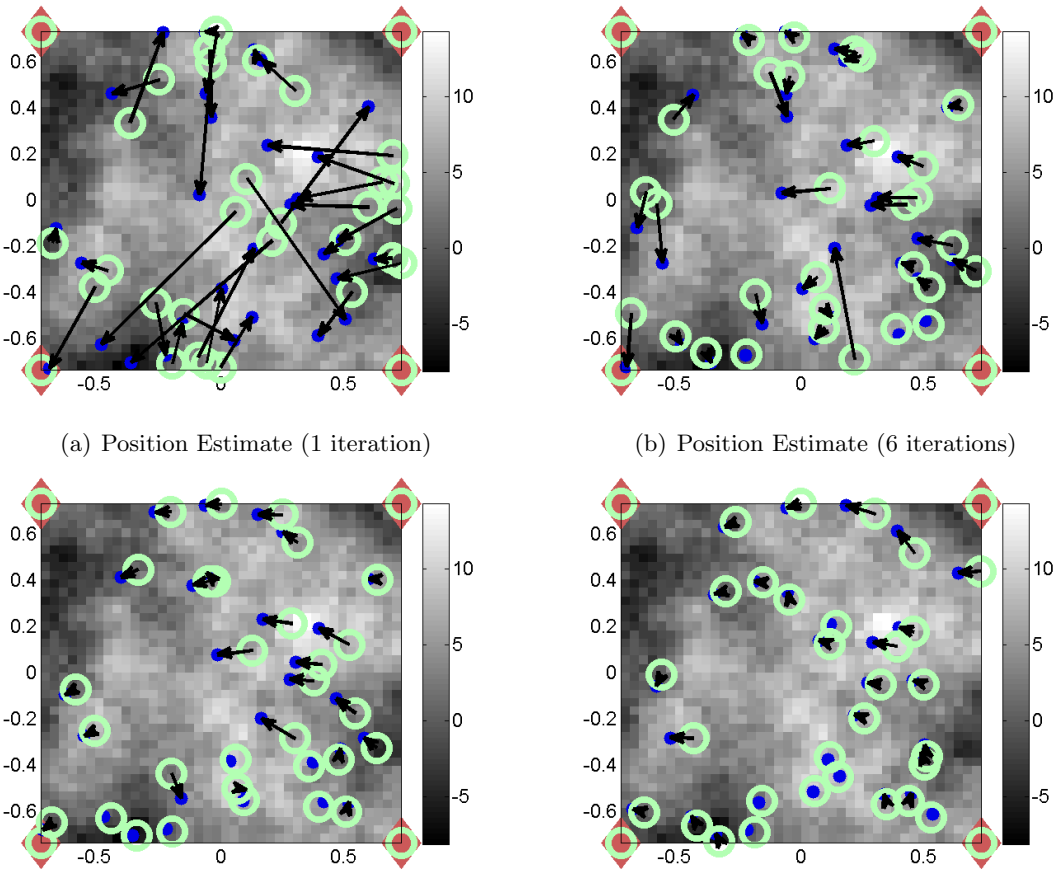
4.5.3 SAL Example

An example simulated network using the SAL algorithm is shown in Figure 24. Snapshots from the first, 6^{th}, 18^{th} and 33^{rd} iterations are shown. The floating node positions from which the measurements were taken are depicted as solid circles, reference nodes as solid diamonds, and the estimated positions as hollow circles. Arrows are drawn from the estimated positions to the true positions for clarity.

Initially, the node locations are completely unknown, so the position distribution for each node is uniform over the entire site, causing the estimated positions to be clustered around the center. After the first iteration (Figure 24(a)), the node's position estimates have shifted towards their true position. As further measurements are made (Figure 24(b),

Algorithm 1 SAL algorithm

At each node n:
Initialize particle positions and weights:
$\boldsymbol{s}_n^{(k)} \sim$ Uniform over network
$w_n^{(k)} = \frac{1}{N_P}$
for each measurement period t **do**
  if Node n is a floating node **then**
    for each neighbor m **do**
      Receive beacon with $\boldsymbol{s}_m$, $\hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m}$
      Form expected RSS measurement $\tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}$ with Eq. (59)
      Adjust weights using Eq. (60)
    end for
    Normalize weights
    Calculate effective number of particles N_{eff}.
    if $N_{\text{eff}} < N_{\text{thr}}$ **then**
      Resample Particles:
      $\boldsymbol{s}_n^{(k)} \sim \sum_k N(\boldsymbol{s}_n^{(k)}, \sigma_s^2)$
      $w_n^{(k)} = \frac{1}{N_P}$
    end if
    Estimate position $\boldsymbol{s}_n = \sum_k w_n^{(k)} \boldsymbol{s}_n^{(k)}$
  else
    Obtain position estimate from external source
  end if
  Broadcast beacon with position estimate $\boldsymbol{s}_n$ to neighboring nodes
end for



(a) Position Estimate (1 iteration)

(b) Position Estimate (6 iterations)

(c) Position Estimate (18 iterations)

(d) Position Estimate (33 iterations)

Figure 24: SAL example

Figure 24(c)), the position estimates become more and more accurate. This process continues as long as location information is desired (Figure 24(d)).

4.5.4 Performance Comparison

In this section we evaluate the performance of the SAL algorithm through Monte Carlo simulations. We evaluate both the steady-state MSE and the convergence rate for the location estimates. For comparison we also show the MSE and convergence rate when the shadowing is assumed to be its average value (0dB) instead of using the shadowing model.

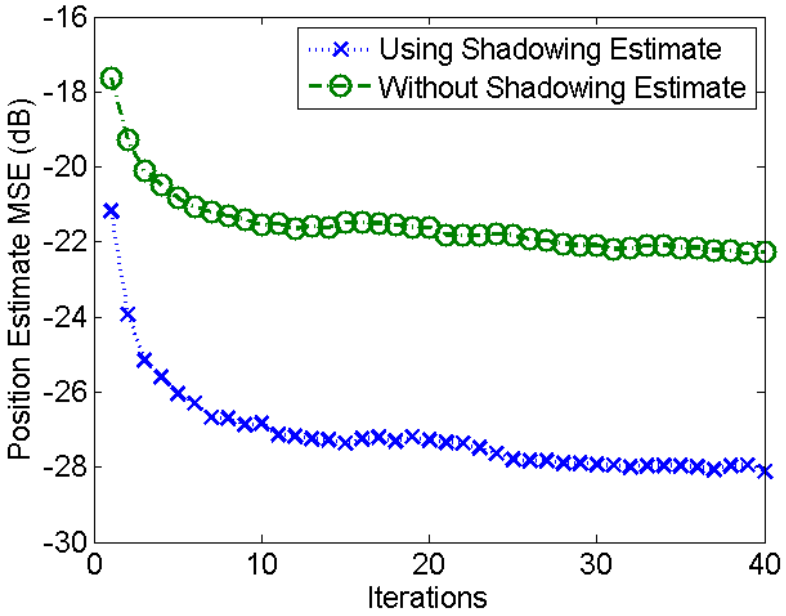
We simulated a network of 20 mobile nodes and 4 static nodes. The mobile nodes were moved according to the random velocity model described in Section 4.3.1. The shadowing was modeled using the NeSh model [50,51] previously described in Chapter 3.3. The spatial loss field had a shadowing variance of σ_η^2 of 100 dB^2/unit range and correlation parameter

κ of 10% of the wireless range. The distance attenuation exponent α was 2. Each node estimated its location using the SAL algorithm with $P = 1000$ particles. We define signal-to-noise ratio (SNR) as the ratio of transmitted power to noise power.

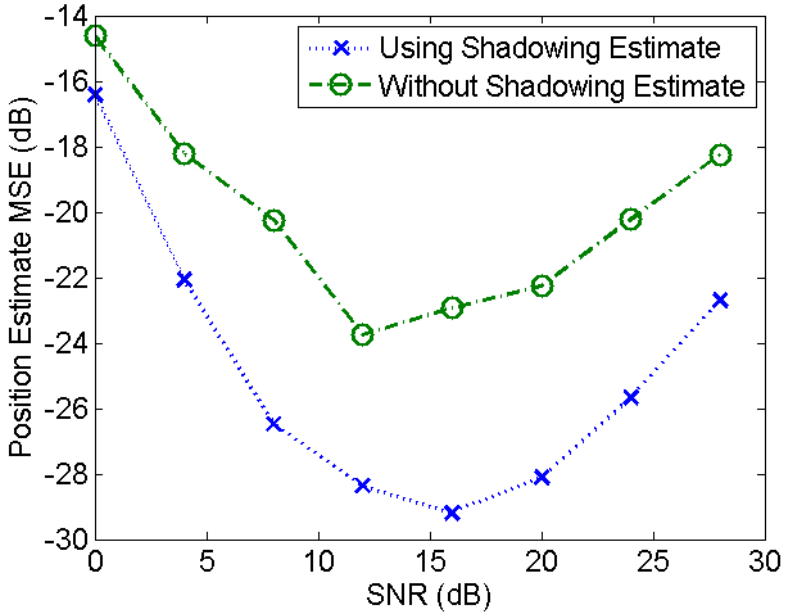
The results are shown in Figure 25. Figure 25(a) shows how the position estimate MSE decreases with each iteration. From the figure it is apparent that the use of the shadowing model does not significantly affect the convergence time of the algorithm, but does significantly reduce the steady-state MSE. The steady-state position estimate MSE is compared for different SNRs in Figure 25(b). From the figure, the use of the shadowing estimate significantly reduces position estimate MSE regardless of SNR. At higher SNRs the position estimate MSE increases because the algorithm is more likely to get stuck in local minima. The SAL algorithm described in this section does not consider the variance of the position estimates of its neighbors when calculating its weights. Since in higher SNR the noise variance σ_v^2 is much lower, the variance in the RSS measurements due to the error in the position estimates of the neighbors begins to dominate. Since the SAL algorithm only considers the noise variance, it underestimates the variance in the RSS measurements, and gets trapped in local minima. This effect can be reduced by increasing the spread parameter σ_s, or increasing the noise variance σ_v^2 used by the algorithm.

4.6 Improved Wireless Localization Performance for Large Networks

In the localization algorithms previously described, nodes estimate their position using a distributed location estimate, without considering the error present in their neighbor's location estimate. The result is that reference nodes and floating nodes both are treated with equal authority for localization. In large networks with relatively few reference nodes, the sheer number of floating nodes with poor measurements degrade the estimate and cause it to converge to the true locations very slowly. We examine the source of this problem by explicitly considering all of the approximations made going from the centralized localization algorithm to the distributed implementations. Then we determine a modification to the distributed implementations that eliminates the approximation responsible for the performance problems.



(a) Position MSE at 20dB SNR



(b) Position MSE after 40 iterations

Figure 25: Simulated performance

4.6.1 Location Estimation Models

In a centralized location estimation algorithm, a single Bayesian filter (such as a Kalman filter or particle filter) would use all the measurements between each of the nodes in the network to simultaneously update the position estimates of all of the floating nodes in each iteration. Consider a system with state $\boldsymbol{\chi}_t = [\boldsymbol{s}_0[t]^T \cdots \boldsymbol{s}_{N-1}[t]^T]^T$ at time t and measurements $\boldsymbol{\zeta}_t = [z_{nm}[t]|n, m \in \mathcal{L}[t]]$, where $\mathcal{L}[t]$ is the set of tuples n, m such that nodes n and m are neighbors at time t. This Bayesian update consists first updating the state probability distribution from time $t-1$ to t, and second of incorporating the likelihood of the current measurement:

$$\mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} = \mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\chi}_{t-1}\right\} \mathrm{P}\left\{\boldsymbol{\chi}_{t-1}|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} \tag{62}$$

$$\mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_t\right\} = \mathrm{P}\left\{\boldsymbol{\zeta}_t|\boldsymbol{\chi}_t\right\} \mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\}. \tag{63}$$

This single filter is very flexible since it can take advantage of correlations between the node positions estimates, and can take into account how uncertainty in the position estimates can affect the measurement likelihood. This flexibility comes at a high cost, however, since it is not easily distributed and computational and memory requirements scale poorly with the size of the network. We instead consider two simplifications of this model which distribute the computation over each node in the network, allowing for a complexity that scales linearly with the number of nodes in the network. In both simplifications we will also assume that node movement is independent, so Eq. (62), simplifies to:

$$\mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} = \mathrm{P}\left\{\boldsymbol{\chi}_{t-1}|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} \prod_m \mathrm{P}\left\{\boldsymbol{s}_m[t]|\boldsymbol{s}_m[t-1]\right\}. \tag{64}$$

In the distributed filter model, each node uses an independent recursive Bayesian filter to estimate its own position using only the position estimates and RSS measurements from its neighbors. This is equivalent to assuming that there is no correlation in the node position estimates and that the measurement likelihood does not depend on the neighbor's position uncertainty. More formally, each node n assumes

$$\mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} = \prod_m \mathrm{P}\left\{\boldsymbol{s}_m[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\}$$

and $\mathrm{P}\left\{\boldsymbol{s}_m[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} = 1$ for $m \neq n$. The measurement update equation then becomes:

$$\mathrm{P}\left\{\boldsymbol{s}_n[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_t\right\} = \mathrm{P}\left\{\boldsymbol{\zeta}_t|\boldsymbol{\chi}_t\right\} \mathrm{P}\left\{\boldsymbol{s}_n[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\}. \tag{65}$$

This effectively treats each of the neighboring nodes as reference nodes. This generally will cause the filter to both underestimate the position uncertainty, and converge to the final solution more slowly. The advantage of this model is that only the node position estimates need to be exchanged between nodes. An extended Kalman filter based on this model has been proposed for localization [69, 85].

In the interlaced filter model, each node in the network estimates its own position based on both the position estimates of its neighbors and the uncertainty in these estimates. This simplification ignores the correlation in position estimates between nodes in the network, but uses the neighbors' position uncertainty to determine the measurement likelihood. Formally, we assume

$$\mathrm{P}\left\{\boldsymbol{\chi}_t|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\} = \prod_m \mathrm{P}\left\{\boldsymbol{s}_m[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\}.$$

The measurement update equation then becomes:

$$\mathrm{P}\left\{\boldsymbol{s}_n[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_t\right\} = \mathrm{P}\left\{\boldsymbol{\zeta}_t|\boldsymbol{\chi}_t\right\} \prod_m \mathrm{P}\left\{\boldsymbol{s}_m[t]|\boldsymbol{\zeta}_0 \cdots \boldsymbol{\zeta}_{t-1}\right\}. \tag{66}$$

If the position estimate error can be assumed to be Gaussian, as is the case when the Kalman filter is employed, this uncertainty can be completely captured by its covariance. This sort of simplification has been described previously in the literature. The simplification of a single Kalman filter to a set of interlaced Kalman filters for general systems is described in greater detail in [88].

4.6.2 Localization with Interlaced Extended Kalman Filters

We can define modification to the localization algorithms described in Section 4.4.2 and Section 4.4.3 to use an interlaced extended Kalman filter. Since the motion of the individual nodes is independent, the only modification to the distributed EKF is the addition of an extra term in the calculation of the covariance of the expected measurement vector $\boldsymbol{S}_m[t]$; replacing Eq. (54) with:

$$\boldsymbol{S}_m[t] = \boldsymbol{H}_m[t]\boldsymbol{E}_m[t|t-1]\boldsymbol{H}_m[t]^H + \boldsymbol{R}_\gamma[t] + \boldsymbol{\Upsilon}_m[t], \tag{67}$$

where $\boldsymbol{\Upsilon} = \text{diag}(\boldsymbol{\upsilon})$, and $\boldsymbol{\upsilon}$ contains the effective variance in the observed measurement due to uncertainty in the position of the corresponding neighboring node. The elements in $\boldsymbol{\upsilon}$ can found to be the diagonal element in the neighboring node n's calculation of $\boldsymbol{H}_n[t]\boldsymbol{E}_n[t|t-1]\boldsymbol{H}_n[t]^H$ that uses node m as a reference node.

More explicitly, the element of $\boldsymbol{\upsilon}$ corresponding to a link between node n and node $m$can be calculated as $\sigma_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t]^2$ where

$$\sigma_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t]^2 = \boldsymbol{j}_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t]^T \boldsymbol{E}_m[t|t-1]\boldsymbol{j}_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t] \tag{68}$$

and

$$\boldsymbol{j}_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t] = -\alpha(\boldsymbol{s}_n[t] - \boldsymbol{s}_m[t])\tilde{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m}[t]^{\frac{\alpha+2}{\alpha}}. \tag{69}$$

4.6.3 Localization with Interlaced Particle Filters

Adapting localization algorithms based on particle filters is more difficult since the posterior probability distribution for the node m in Eq. (66) is approximated by the distribution of node m's particles $\boldsymbol{s}_m^k$. Communicating the state and weight the particles of every node at every iteration would introduce a prohibitive amount of communication overhead. Instead, we approximate the covariance of the position estimate with the sample covariance of the node m's particles:

$$\boldsymbol{E}_n = \frac{\sum_{k=0}^{N_P-1} w_n^{(k)}(\boldsymbol{s}_n^{(k)} - \boldsymbol{s}_n)(\boldsymbol{s}_n^{(k)} - \boldsymbol{s}_n)^T}{1 - \sum_{k=0}^{N_P-1} {w_n^{(k)}}^2}. \tag{70}$$

In this interlaced particle filter for localization each floating node n receive beacons from each neighbor m containing their neighbor's position estimate's covariance $\boldsymbol{E}_m$ at each measurement interval t. As in the distributed particle filter, node n calculates the expected measurement $\tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}$ for each particle k. Since this expected measurement depends on the position of the neighboring node m, some of the uncertainty in the neighboring node's position will propagate through to expected measurement. The measurement is a nonlinear function of position, so the actual effect of the position uncertainty cannot be found analytically. We use the Jacobian of the measurement function, as a first-order approximation to model the propagation of the position uncertainty to this expected measurement. This

Jacobian is found to be:

$$\boldsymbol{j}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} = -\alpha(\boldsymbol{s}_n^{(k)} - \boldsymbol{s}_m)\tilde{z}^{\frac{\alpha+2}{\alpha}}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}. \tag{71}$$

Using this, we can approximate the variance in expected measurement due to the uncertainty in the position of neighboring node m as:

$$\sigma^2_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} = \boldsymbol{j}^T_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} \boldsymbol{E}_m \boldsymbol{j}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}. \tag{72}$$

The particle weights can then be updated as

$$w_n^{(k)}[t+1] = w_n^{(k)}[t] \prod_m q[n,k,m], \tag{73}$$

where $q[n,k,m]$ is the weight adjustment for particle k from the measurement from node m. This weight adjustment should be proportional to

$$\mathrm{P}\left\{ z_{\boldsymbol{s}_n,\boldsymbol{s}_m} = \tilde{z}_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m} | \hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m} \right\}.$$

Since we are assuming that both the measurement noise γ and the uncertainty in the expected measurement are Gaussian with the same mean, the weight adjustment $q[n,k,m]$ in Eq. (73) should be the Gaussian probability density function with variance $\sigma_\mu[n,k,m]^2$:

$$q[n,k,m] = \frac{1}{\sqrt{2\pi\sigma_\mu[n,k,m]^2}} e^{-\frac{\left(\hat{z}_{\boldsymbol{s}_n,\boldsymbol{s}_m} - \tilde{z}_{\boldsymbol{s}_n^{(k)},m}\right)^2}{2\sigma_\mu[n,k,m]^2}}, \tag{74}$$

where

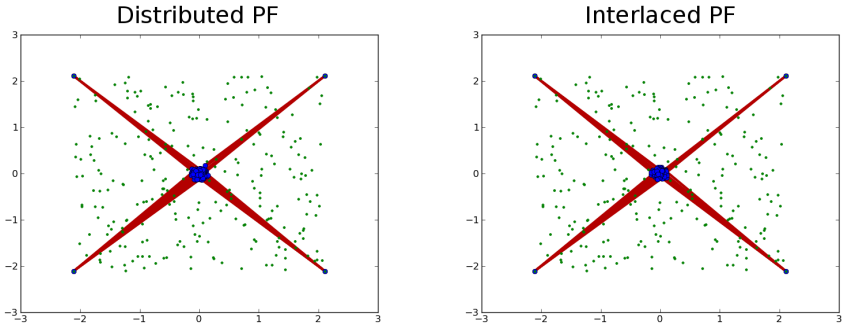
$$\sigma_\mu[n,k,m]^2 = \sigma_\gamma^2 + \sigma^2_{\boldsymbol{s}_n^{(k)},\boldsymbol{s}_m}.$$

4.6.4 Performance Comparison

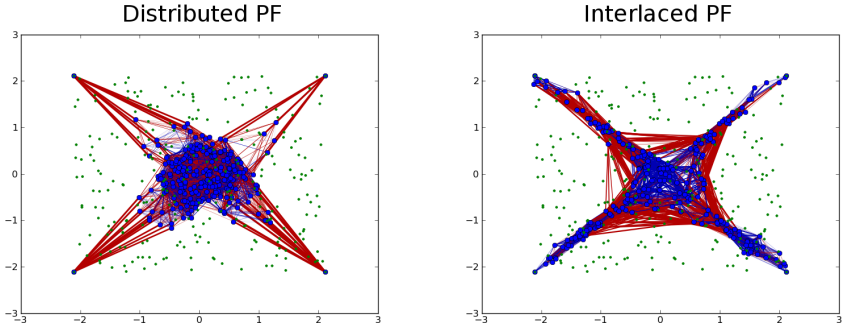
We compare the performance of the interlaced extended Kalman filter presented in Section 4.6.2, the interlaced particle filter presented in Section 4.6.3, the distributed extended Kalman filter algorithm from Section 4.4.3 and the distributed particle filter algorithm from Section 4.5 through simulation. We simulate a network of nodes in a square area with 4 reference nodes at each of the corners of the network area. The network area was assumed to have negligible (0dB) shadowing. Nodes were assumed to have a range of 1 unit, and the size of the network was adjusted such that the average node density was 14 nodes per unit area, or

14π neighbors per node. Nodes moved according to the random velocity model described in Section 4.3.1, and simulated RSS or distance measurements were taken every 0.001 time units and used for localization. The simulation was run for 300 measurement intervals. The position MSE was measured at each interval, and then averaged over 50 simulations.

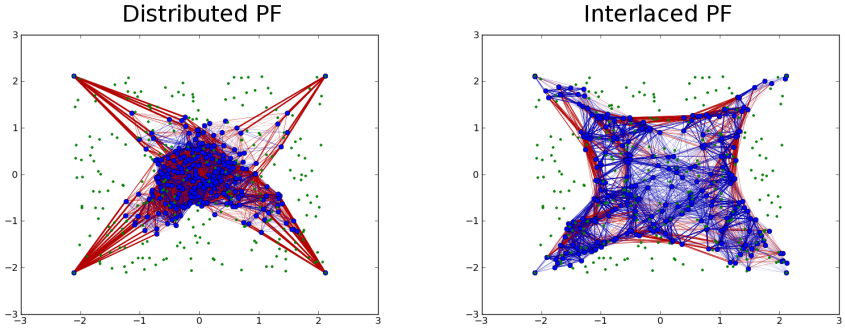
Figure 26 shows an example comparing the distributed particle filter with the interlaced particle filter in a network of 250 nodes after 0, 30, 120 and 300 measurement and localization iterations. There are 4 reference nodes, each located at one of the corners of the network. The actual positions of the nodes are represented by the small circles, and the current position estimate is shown by the larger circles. The lines represent links between nodes, and the thickness of the line represents the magnitude of the difference between the true distance of this link and distance using the current position estimate. The two filters start with roughly the same initial estimates, as shown in Figure 26(a). In the distributed particle filter, measurements from all the neighbors are given equal weight, so the position estimates are "pushed" outward from the center in almost a random direction depending on the initial position estimate and into a local minimum. In contrast, the interlaced particle filter, nodes are almost unaffected by measurements to neighbors that have poor location estimates, so nodes directly connected to reference nodes are "pulled" toward them, forming a sort of 'X' shape. These effects are shown in the 30^{th} measurement iteration, in Figure 26(b). In the distributed particle filter, the position estimates are slowly pulled to their proper position, but they have to go against the established local minimum, so progress is slow. In the interlaced particle filter, the position estimates for the nodes directly connected to reference nodes slowly improve, and begin to pull their neighbors, and their neighbor's neighbors towards the reference nodes. Eventually, nodes indirectly connected to reference nodes are pulled towards more than one reference node, causing them to be pulled towards a point between the reference nodes. This causes the position estimates to begin to fill out the 'X' shape. These effects are appear by the 120^{th} iteration, shown in Figure 26(c). Eventually the interlaced particle filter converges to a good estimate of the nodes' position, while the distributed particle filter is still trying to escape from its local minimum, shown in Figure 26(d).



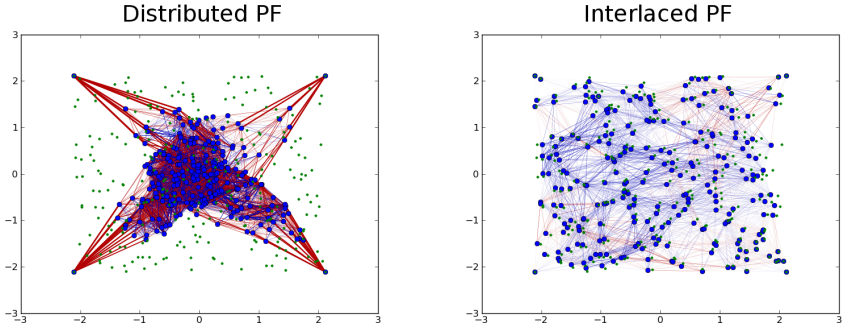
(a) Iteration 0



(b) Iteration 30

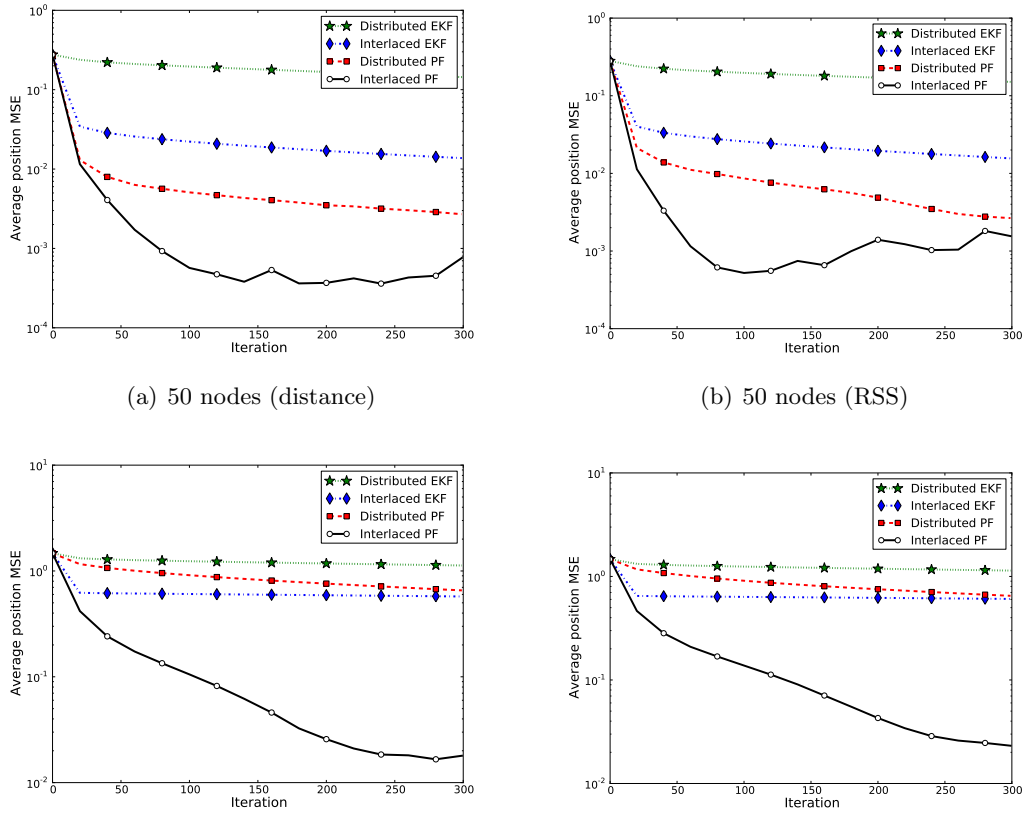


(c) Iteration 120



(d) Iteration 300

Figure 26: Improved localization example



(a) 50 nodes (distance)

(b) 50 nodes (RSS)

(c) 250 nodes (distance)

(d) 250 nodes (RSS)

Figure 27: Position MSE convergence

The average position estimate MSE for the first 300 localization iterations are shown in Figure 27. Figure 27(a) shows the position MSE for networks of 50 nodes using distance measurements, and Figure 27(b) shows the position MSE using RSS measurements. These figures show that algorithms using particle filters perform much better than algorithms based on Kalman filters. Indeed, the distributed EKF offers a negligible improvement over its initial guess. The distributed particle filter has an initial rapid improvement in MSE, but then only offers slow improvements over each iteration as it tries to escape its local minimum. The interlaced particle filter has a position estimate MSE that is strictly less than that provided by the other algorithms, and quickly converges to its minimum MSE.

In larger networks, these results change. Figure 27(c) and Figure 27(d) show the corresponding MSE for networks of 250 nodes. While the distributed EKF still offers only a negligible improvement, the distributed particle filter performs much more poorly. With

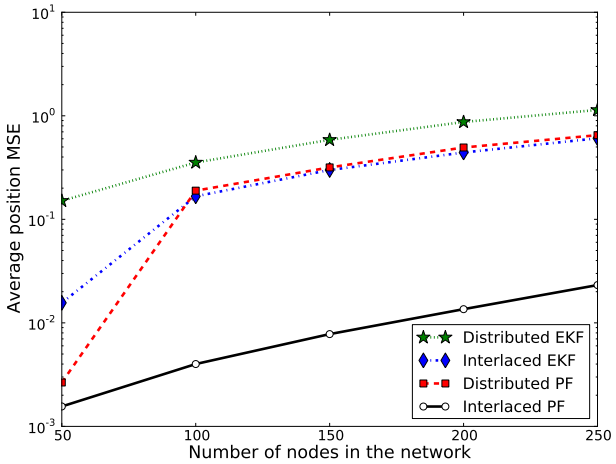


Figure 28: Position MSE after 300 iterations (RSS)

such a large number of nodes, it gets stuck in a local minimum at a much higher MSE, and escapes it much more slowly. For this reason, the two interlaced filters converge much faster than the distributed methods in larger networks. The interlaced EKF, which relies on a Gaussian approximation for the position estimate distribution, is unable to match the the MSE performance of a particle filter, whose particles can more closely approximate the true position estimate distribution. For this reason, the two particle filter methods will have a lower steady-state MSE. The interlaced particle filter both converges much faster and has a lower steady-state MSE than the other methods.

If we only consider how the position estimate MSE scales with network size, the interlaced particle filter also outperforms the other methods. The position estimate MSE using RSS measurements after 300 iterations is shown in Figure 28. In this figure, the position MSE achieved by the interlaced particle filter increases much slower with network size than the other methods.

4.7 Concluding Remarks

We have presented two localization algorithms. One algorithm is designed for networks without shadowing and performs localization and location tracking using both RSS and acceleration measurements. The other algorithm, termed 'SAL,' is designed specifically for networks with strong shadowing, and uses a shadowing model of the network area to

improve localization and tracking performance. We evaluated these two algorithms through simulation and demonstrated their utility within their respective operating environments. We have also described how they can be modified to increase localization accuracy and algorithm scalability at a cost of a small amount of additional communication overhead.

CHAPTER V

CONCLUSIONS AND FUTURE WORKS

In this work we have shown how Bayesian filtering techniques can be used to improve the performance of algorithms for time synchronization, RF tomography, and localization in wireless networks. More specifically, this work has addressed the following:

- Network Time Synchronization
  - Introduced a new clock model for time synchronization in wireless networks
  - Presented novel time synchronization algorithm ACES based on Kalman filter
  - Analyzed ACES through both analysis and simulation to show it offers both high accuracy and efficiency
- RF Tomography
  - Described how wireless networks can sense the objects in the network area using only RSS measurements.
  - Introduced a new algorithm RETINA, capable of joint detection and tracking of both stationary and moving objects in the environment
  - Compared RETINA with existing methods through simulation
  - Introduced novel shadowing model, the Inverse Area Ellipse model
  - Described our RF tomography testbed and field test
  - Showed that the Inverse Area Ellipse model is more consistent with the channels measured during the field test than other existing models
- Wireless Localization
  - Presented a new hybrid algorithm capable of combining RSS and acceleration measurements to improve the performance of distributed self-localization in mobile networks

- Demonstrated that this hybrid algorithm outperforms existing algorithms which are unable to use acceleration measurements
- Presented SAL, a novel algorithm which uses a model of the wireless channel in the network along with RSS measurements to increase self-localization accuracy in wireless networks
- Demonstrated through simulation that SAL outperforms existing algorithms which cannot use channel models along with RSS measurements
- Investigated a modification to distributed localization algorithms to improve performance in large networks with a small increase in communication overhead
- Demonstrated that this modification improves the convergence time of localization algorithms through simulation

There is still an enormous amount of work that can be done to further improve the methods discussed in this work. For the time synchronization problem, algorithms should consider incorporating additional information such as that from temperature or humidity sensors to produce models for clocks in these resource constrained networks that can achieve higher accuracy with even less frequent synchronization intervals. Additionally, work on RF tomography has only begun. Although we have presented a model for radio propagation that is consistent with measurements in our testbed, additional measurements of the shadowing present in different environments are essential to validating this model. Also, more complicated channel models should be investigated to increase both the resolution and robustness of RF tomography. Further research on reconstruction algorithms is also needed. Although RETINA can detect and track both stationary and mobile objects even when the nodes making the RSS measurements are mobile, its accuracy tracking mobile objects decreases with the speed of the measuring nodes. Further research is needed to identify and correct for the source of this error. More work is also needed to characterize how robust RETINA and other reconstruction algorithms are to errors in the location information of the measuring nodes. Future works could also investigate combining self-localization algorithms such as SAL, with channel modeling algorithms such as RETINA. Such joint

algorithms would greatly increase the number of applications that can take advantage of these powerful techniques.

REFERENCES

[1] Veitch, D., Babu, S., and Pàsztor, A., "Robust synchronization of software clocks across the internet," in *Proc. ACM SIGCOMM IMC*, pp. 219–232, Oct. 2004.

[2] Wilson, J. and Patwari, N., "See-through walls: Motion tracking using variance-based radio tomography networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 5, pp. 612–621, 2011.

[3] Fomin, V. N., *Optimal filtering*. Kluwer Academic Publishers, 1998.

[4] Diard, J., Bessire, P., and Mazer, E., "A survey of probabilistic models, using the bayesian programming methodology as a unifying framework," in *Proc. CIRAS*, 2003.

[5] Haykin, S., *Kalman Filters*. John Wiley & Sons, Inc., 2002.

[6] Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T., "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, 2002.

[7] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E., "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393 – 422, 2002.

[8] Hamilton, B. R., "Low-overhead opportunistic routing for wireless ad-hoc and sensor networks in a fading evaluating environment," Master's thesis, Georgia Institute of Technology, 2007.

[9] Hamilton, B. R. and Ma, X., "Noncooperative routing with cooperative diversity," in *Proc. International Conference on Communications*, May 2007.

[10] Biswas, S. and Morris, R., "ExOR: Opportunistic multi-hop routing for wireless networks," in *Proc. ACM SIGCOMM*, pp. 133–144, ACM, Aug. 2005.

[11] Karp, B. and Kung, H. T., "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. IEEE/ACM MobiCom*, pp. 243–254, Aug. 2000.

[12] Kranakis, E., Singh, H., and Urrutia, J., "Compass routing on geometric networks," in *Proc. Canadian Conference on Computational Geometry*, pp. 51–54, 1999.

[13] Johnson, D. and Maltz, D., "Dynamic source routing in ad-hoc wireless networks," in *Proc. of SIGCOMM*, Aug. 1996.

[14] Perkins, C. E. and Royer, E. M., "Ad-hoc on-demand distance vector routing," in *Proc. IEEE WMCSA*, vol. 3, pp. 90–100, Aug. 1999.

[15] Sundararaman, B., Buy, U., and Kshemkalyani, A., "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks*, vol. 3, Feb. 2005.

[16] Hamilton, B. R., Ma, X., Zhao, Q., and Xu, J., "ACES: adaptive clock estimation and synchronization using Kalman filtering," in *Proc. ACM MOBICOM*, (New York, NY, USA), pp. 152–162, ACM, 2008.

[17] Mills, D. L., "Internet time synchronization: the network time protocol," *IEEE Trans. Commun.*, vol. 39, pp. 1482–1493, Oct. 1991.

[18] Mills, D. L., "Improved algorithms for synchronizating computer network clocks," in *IEEE/ACM Trans. Networking*, vol. 3, pp. 245–254, June 1995.

[19] Liao, C., Martonosi, M., and Clark, D. W., "Experience with an adaptive globally-synchronizing clock algorithm," in *ACM Symposium on Parallel Algorithms and Architectures*, pp. 106–114, 1999.

[20] Pàsztor, A. and Veitch, D., "PC based precision timing without GPS," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 30, pp. 1–10, June 2002.

[21] "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," IEEE Std. 1588-2002, 2002.

[22] Moon, S. B., Skelly, P., and Towsley, D., "Estimation and removal of clock skew from network delay measurements," in *INFOCOM 1999*, vol. 1, pp. 227–234, Mar. 1999.

[23] Zhang, L., Liu, Z., and Xia, C. H., "Clock synchronization algorithms for network measurements," in *INFOCOM 2002*, vol. 1, pp. 160–169, 2002.

[24] Pottie, G. and Kaiser, W., "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, pp. 51–58, May 2000.

[25] Vig, J. R., "Introduction to quartz frequency standards," Tech. Rep. SLCET-TR-92-1 (Rev. 1), Army Research Laboratory, Oct. 1992.

[26] Phillips, J. and Kundert, K., "Noise in mixers, oscillators, samplers, and logic: An introduction to cyclostationary noise," in *IEEE CICC*, pp. 431–438, May 2000.

[27] Romer, K., "Time synchronization in ad hoc networks," in *ACM MobiHoc*, Oct. 2001.

[28] Ganeriwal, S., Kumar, R., and Srivastava, M., "Timing-sync protocol in sensor networks," in *ACM SENSYS*, Nov. 2003.

[29] Maróti, M., Kusy, B., Simon, G., and Lédeczi, Á., "The flooding time synchronization protocol," in *Proc. SenSys*, (New York, NY, USA), pp. 39–49, ACM, 2004.

[30] Sichitiu, M. and Veerarittiphan, C., "Simple, accurate time synchronization for wireless sensor networks," in *IEEE WCNC*, 2003.

[31] Li, Q. and Rus, D., "Global clock synchronization in sensor networks," in *IEEE INFOCOM*, Mar. 2004.

[32] Zhou, D. and Lai, T., "A scalable and adaptive clock synchronization protocol for IEEE 802.11-based multihop ad hoc networks," in *IEEE Mobile Adhoc and Sensor Systems*, Nov. 2005.

[33] Elson, J., Girod, L., and Estrin, D., "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, 2002.

[34] Mock, M., Frings, R., Nett, E., and Trikaliotis, S., "Continuous clock synchronization in wireless real-time applications," in *IEEE SRDS*, pp. 125–133, Oct. 2000.

[35] PalChaudhuri, S., Saha, A., and Johnson, D., "Adaptive clock synchronization in sensor networks," in *IEEE IPSN*, Apr. 2004.

[36] Su, W. and Akyildiz, I., "Time-diffusion synchronization protocols for sensor networks," *IEEE/ACM Trans. Netw.*, 2005.

[37] Wu, Y., Chaudhari, Q., and Serpedin, E., "Clock synchronization of wireless sensor networks," *IEEE Signal Processing Magazine*, pp. 124–138, Jan. 2011.

[38] Noh, K., Chaudhari, Q. M., Serpedic, E., and Suter, B. W., "Novel clock phase offset and skew estimation using two-way timing message exchanges for wireless sensor networks," *IEEE Trans. Commun.*, vol. 55, Apr. 2007.

[39] Leng, M. and Wu, Y., "On clock synchronization algorithms for wireless sensor networks under unknown delay," *IEEE Trans. Veh. Technol.*, vol. 59, pp. 182–190, Jan. 2010.

[40] Kim, K. S. and Lee, B. G., "KALP: a Kalman filter-based adaptive clock method with low-pass prefiltering for packet networks use," *IEEE Trans. Commun.*, vol. 48, pp. 1217–1225, July 2000.

[41] Bletsas, A., "Evaluation of Kalman filtering for network time keeping," *IEEE Trans. Ultrason., Ferroelectr., Freq. Control*, vol. 52, Sept. 2005.

[42] Auler, L. and d'Amore, R., "Adaptive Kalman filter for time synchronization over packet-switched networks (an heuristic approach)," in *IEEE COMSWARE*, Jan. 2007.

[43] "Clock Oscillator Stability." Cardinal Components Inc. Applications Brief No. A.N. 1006, www.cardinalxtal.com/docs/notes.

[44] Jones, R. H. and Boadi-Boateng, F., "Unequally spaced longitudinal data with AR(1) serial correlation," *Biometrics*, vol. 47, no. 1, pp. 161–175, 1991.

[45] Kim, H., Ma, X., and Hamilton, B. R., "Modeling and tracking clocks with time-varying skews using information criteria," in *Proc. IEEE MILCOM*, pp. 528–533, 2010.

[46] Kim, H., Ma, X., and Hamilton, B. R., "Tracking low-precision clocks with time-varying drifts using Kalman filtering," *IEEE/ACM Trans. Netw.*, "accepted".

[47] Stüber, G. L., *Principles of Mobile Communications*. Kluwer Academic Publishers, 2nd ed., 2001.

[48] Graziosi, F. and Santucci, F., "A general correlation model for shadow fading in mobile radio systems," *IEEE Commun. Lett.*, vol. 6, no. 3, pp. 102–104, 2002.

[49] Gudmundson, M., "Correlation model for shadow fading in mobile radio systems," *Electronics Letters*, vol. 27, no. 23, pp. 2145–2146, 1991.

[50] Agrawal, P. and Patwari, N., "Correlated link shadow fading in multi-hop wireless networks," *IEEE Trans. Wireless Commun.*, vol. 8, pp. 4024–4036, Aug. 2009.

[51] Patwari, N. and Agrawal, P., "NeSh: A joint shadowing model for links in a multi-hop network," in *Proc. IEEE ICASSP*, pp. 2873–2876, 2008.

[52] Wilson, J. and Patwari, N., "Radio tomographic imaging with wireless networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 5, pp. 621–632, 2010.

[53] Mostofi, Y., "Compressive cooperative sensing and mapping in mobile networks," *Mobile Computing, IEEE Transactions on*, vol. 10, pp. 1769 –1784, Dec. 2011.

[54] Herman, G. T., *Fundamentals of Computerized Tomography: Image Reconstruction from Projections*. Springer, 2nd ed., 2009.

[55] Youssef, M., Mah, M., and Agrawala, A., "Challenges: device-free passive localization for wireless environments," in *Proc. ACM MOBICOM*, pp. 222–229, 2007.

[56] Monte, L. L., Patton, L. K., and Wicks, M. C., "Direct-path mitigation for underground imaging in RF tomography," in *Proc. ICEAA*, pp. 27–30, 2010.

[57] Wicks, M. C., "RF tomography with application to ground penetrating radar," in *Proc. ACSSC*, pp. 2017–2022, 2007.

[58] Wilson, J. and Patwari, N., "Regularization methods for radio tomographic imaging," in *Proc. Virginia Tech Wireless Symposium*, June 2009.

[59] Hamilton, B. R., Ma, X., Baxley, R. J., and Matechik, S. M., "Radio frequency tomography in mobile networks," in *Proc. SSP*, (submitted).

[60] Hamilton, B. R., Ma, X., Baxley, R. J., and Matechik, S. M., "Propagation modeling for radio frequency tomography in wireless networks," in *Proc. ACM MOBICOM*, (submitted).

[61] "The universal software radio peripheral (USRP)." Ettus Research LLC.

[62] Yu, Z., Baxley, R. J., Walkenhorst, B. T., and Zhou, G. T., "Channel sounding waveforms design for asynchronous multiuser MIMO systems." Technical Report, Georgia Tech Research Institute, 2012.

[63] Patwari, N., Ash, J. N., Kyperountas, S., Hero, A. O. I., Moses, R. L., and Correal, N. S., "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 22, pp. 54–69, July 2005.

[64] Mao, G., Fidan, B., and Anderson, B. D., "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[65] Guvenc, I. and Chong, C.-C., "A survey on TOA based wireless localization and NLOS mitigation techniques," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 3, pp. 107–124, 2009.

[66] Tran, D. A. and Nguyen, T., "Localization in wireless sensor networks based on support vector machines," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 7, pp. 981–994, 2008.

[67] Hu, L. and Evans, D., "Localization for mobile sensor networks," in *Proc. IEEE/ACM MobiCom*, (New York, NY, USA), pp. 45–57, ACM, 2004.

[68] Baggio, A. and Langendoen, K., "Monte carlo localization for mobile wireless sensor networks," *Ad Hoc Networks*, vol. 6, pp. 718–733, July 2007.

[69] Savvides, A., Park, H., and Srivastava, M. B., "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proc. ACM WSNA*, (New York, NY, USA), pp. 112–121, ACM, 2002.

[70] Biswas, P. and Ye, Y., "Semidefinite programming for ad hoc wireless sensor network localization," in *Proc. IEEE/ACM IPSN*, pp. 46–54, Apr. 2004.

[71] Chang, C.-H. and Liao, W., "Revisiting relative location estimation in wireless sensor networks," in *Proc. IEEE ICC*, pp. 1–5, June 2009.

[72] Savvides, A., Garber, W., Adlakha, S., Moses, R., and Srivastava, M. B., "On the error characteristics of multihop node localization in ad-hoc sensor networks," in *Proc. IEEE/ACM IPSN*, vol. 2634, p. 555, 2003.

[73] Savvides, A., Garber, W. L., Moses, R. L., and Srivastava, M. B., "An analysis of error inducing parameters in multihop sensor node localization," *IEEE Trans. Mobile Comput.*, vol. 4, pp. 567–577, Nov. 2005.

[74] Patwari, N., Hero, A. O. I., Perkins, M., Correal, N. S., and O'Dea, R. J., "Relative location estimation in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2137–2148, 2003.

[75] Elnahrawy, E., Li, X., and Martin, R. P., "The limits of localization using signal strength: a comparative study," in *Proc. IEEE SECON*, pp. 406–414, 2004.

[76] Patwari, N. and Agrawal, P., "Effects of correlated shadowing: connectivity, localization, and RF tomography," in *Proc. IEEE/ACM IPSN*, pp. 82–93, Apr. 22–24, 2008.

[77] Smailagic, A. and Kogan, D., "Location sensing and privacy in a context-aware computing environment," *IEEE Wireless Commun. Mag.*, vol. 9, no. 5, pp. 10–17, 2002.

[78] Bahl, P. and Padmanabhan, V. N., "RADAR: an in-building RF-based user location and tracking system," in *Proc. IEEE INFOCOM*, vol. 2, pp. 775–784, 2000.

[79] Seshadri, V., Zaruba, G. V., and Huber, M., "A bayesian sampling approach to in-door localization of wireless devices using received signal strength indication," in *Proc. IEEE PerCom*, pp. 75–84, Mar. 8–12, 2005.

[80] Morelli, C., Nicoli, M., Rampa, V., Spagnolini, U., and Alippi, C., "Particle filters for RSS-based localization in wireless sensor networks: an experimental study," in *Proc. IEEE ICASSP*, vol. 4, 2006.

[81] Hightower, J. and Borriello, G., "Particle filters for location estimation in ubiquitous computing: A case study," in *UbiComp*, vol. 3205, pp. 88–106, 2004.

[82] Camp, T., Boleng, J., and Davies, V., "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, pp. 483–502, Sept. 2002.

[83] Yoon, J., Liu, M., and Noble, B., "Sound mobility models," in *Mobicom*, Sept. 2003.

[84] Hamilton, B. R., Ma, X., Baxley, R. J., and Walkenhorst, B., "Node localization and tracking using distance and acceleration measurements," in *Proc. CIP*, June 2010.

[85] Savvides, A., Park, H., and Srivastava, M. B., "The n-hop multilateration primitive for node localization problems," *Mobile Network Applications*, vol. 8, no. 4, pp. 443–451, 2003.

[86] Tichavsky, P., Muravchik, C. H., and Nehorai, A., "Posterior cramer-rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Process.*, vol. 46, pp. 1386–1396, May 1998.

[87] Hamilton, B. R., Ma, X., and Baxley, R. J., "SAL: Shadowing assisted localization," in *Proc. SSP*, June 2011.

[88] Glielmo, L., Setola, R., , and Vasca, F., "An interlaced extended Kalman filter," *IEEE Trans. Autom. Control*, vol. 44, pp. 1546–1549, Aug. 1999.